REPORT DOCUMENTATION PAGE

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
19-06-2009	Final	March 16, 2006 - June 30, 2009

4. TITLE AND SUBTITLE

ONR C-IED STIFLE (Stigmergic Tracking of IED Factories, Locations and Events)

5a. CONTRACT NUMBER

5b. GRANT NUMBER

N00014-06-1-0467

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Brueckner, Sven A.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

TechTeam Government Solutions, Inc.
3520 Green Ct. Ste. 250
Ann Arbor, MI 48105

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

The Office of Naval Research
Counter-IED S&T Dept. (CODE 30)
c/o CAPT Mark A. Stoffel
875 N. Randolph St., RM 1425
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)

ONR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for Public Release; distribution is Unlimited.

20090710276

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The STIFLE project, funded by the ONR Counter-IED Basic Research program, has three major objectives, which are reflected in the three main tracks of our project execution:
1. "Enhanced Representations" Track: Extend the predictive polyagent modeling construct to include explicit reasoning over task execution by individuals and groups
2. "Model Analysis" Track: Develop theoretical, formal and experimental analysis tools and methods to characterize and influence the dynamics of predictive polyagent models
3. "IED Prediction Prototype" Track: Apply the extended modeling and analysis capabilities to the problem of IED prediction and forensics
Towards the first objective, we are collaborating with Prof. Keith Decker (University of Delaware) to integrate our polyagent modeling approach within the TAEMS framework, a formal representation and reasoning mechanism for hierarchical task networks

15. SUBJECT TERMS

improvised explosive devices, IEDs, polyagents, swarming, scheduling, prediction, self-organization

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Ted Belding
U	U	U	SAR	47	**19b. TELEPHONE NUMBER** (Include area code) 734-302-4667

1. **Title: "ONR C-IED STIFLE (Stigmergic Tracking of IED Factories, Locations and Events)"**
2. **Prime Offeror:** TechTeam Government Solutions, Inc. (previously Altarum Institute)
3. **Subcontractors:**
   (a) Dr. Keith Decker, University of Delaware
   (b) Dr. Robert Savit, Savit Research Associates and University of Michigan
4. **Period of Performance:** March 16, 2006 to June 30, 2009
5. **Submitted by:** Dr. Sven Brueckner, 734-302-4683, sven.brueckner@newvectors.net
6. **Business Contact:** Annie Smallwood, 734-302-4736, annie.smallwood@newvectors.net

7. Background/Scope of Effort

Factories, deployment locations, and detonation events associated with IEDs (improvised ex-plosive devices) are not random, but are constrained by a wide range of environmental features, including the locations and movements of likely targets, security patrols, friendly civilians, military and police facilities, and other factors that are subject to change over time. For instance, experience shows that disciplined patrols can reduce IED emplacement.

STIFLE (Stigmergic Tracking of IED Factories, Locations, and Events) allows Navy and Marine forces to visualize the effects such features will have on IED placement. It uses a multi-agent simulation to model the interaction of insurgents, targets, patrols, and other factors of the battlespace that affect IED manufacturing, placement, and distribution. This allows prediction of likely IED factory areas, locations, and events.

8. Summary/Abstract

The STIFLE project, funded by the ONR Counter-IED Basic Research program, has three major objectives, which are reflected in the three main tracks of our project execution:

1. "Enhanced Representations" Track: Extend the predictive polyagent modeling construct to include explicit reasoning over task execution by individuals and groups

2. "Model Analysis" Track: Develop theoretical, formal and experimental analysis tools and methods to characterize and influence the dynamics of predictive polyagent models

3. "IED Prediction Prototype" Track: Apply the extended modeling and analysis capabilities to the problem of IED prediction and forensics

Towards the first objective, collaborated with Prof. Keith Decker (University of Delaware) to integrate our polyagent modeling approach within the TAEMS framework, a formal representation and reasoning mechanism for hierarchical task networks. Together with Prof. Bob Savit (University of Michigan), we explored various approaches to formally describe and analyze our predictive polyagent models in support of the second objective. Finally, our development and experimental analysis of alternative polyagent models of IED emplacement (based on initial models and a framework supported by the DARPA RAID adversarial reasoning module) supported the third objective.

9. Technical Contents and Accomplishments

The accomplishments for each of the three project tracks are described in the following sections.

9.1 IED Prediction Prototype Track

We developed a baseline IED emplacement risk prediction model that under ONR C-IED funding, building on the software infrastructure initially developed in the DARPA RAID program under the ARM-N module. In summary, the model deploys swarms of fine-grained agents that move probabilistically on a multi-pheromone landscape. The agents carry a parameterized personality model that determines their response to particular pheromone flavors. These flavors are representative of the recent (relative to the overall amount of history available to the

Figure 1. Baseline IED prediction model integrates over all events in a given time window.

prediction engine) local presence of Blue convoys (attractive), patrols (repulsive), or IED events (attractive) – see Figure 1. An additional flavor used by these agents conveys a statistical long-term assessment of the level of IED threat (attractive) at a particular location. The spatial distribution of the agent population that results from the individual integration of these pheromone flavors in the agent's personality model is interpreted as a map of the risk of IED emplacement.
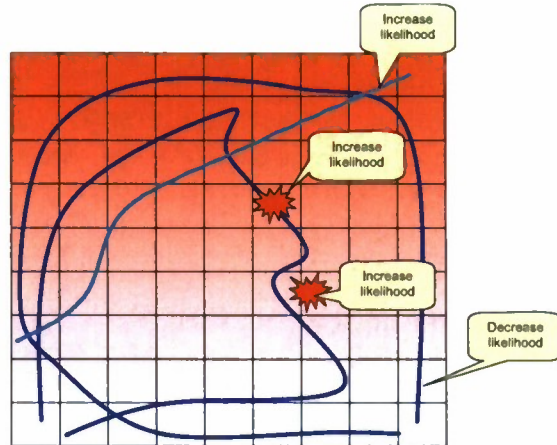
We call this model of agent-based IED prediction the "synchronic" model, because it does not incorporate a polyagent-based prediction of the evolution of the world. Instead it is based on probabilistic agents that are synchronously reasoning about the same world state. This model is our baseline to measure improvements that we may achieve in the deployment of more sophisticated polyagent models and techniques developed with the support of the STIFLE project.

In this period of performance, we have performed some experiments with the synchronic baseline model, using real-world and synthetically generated data from the DARPA RAID program. In the following, we first discuss the metric that we are applying and then we present some results.

9.1.1 Normalized Coverage Ratio & Receiver Operating Characteristics (ROC) Curves

We order a sequence of past IED events chronologically, and choose a point to divide it into training and test data. All of the training data comes before all of the test data. Varying the point of division enables us to explore how our accuracy varies with the amount of training data we have available. Note that the entire body of historical data, both the training segment and the test segment, implicitly includes information on Blue movements, since we only discover IEDs in areas that Blue has visited.

We apply the baseline system to the training data, yielding a threat map that may be compared with a mountainous landscape. Figure 2 shows a notional threat landscape.

We turn this landscape into threat regions by applying a threshold and reporting the contours of the landscape at that threshold. Very high thresholds yield
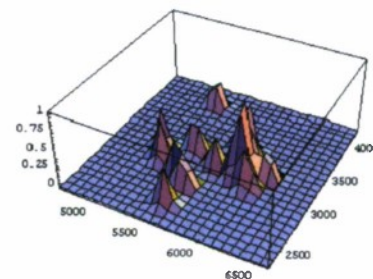

Figure 2. Notional Threat Landscape.

only a few regions (Figure 3). As the threshold drops, the number of regions will vary, increasing as new peaks are exposed and decreasing as previously distinct peaks merge (Figure 4), until at threshold 0 we have only one threat region, covering the entire playbox. For any given threshold, we can compute the coverage percentage, p_c, the percentage of the playbox occupied by threat regions. This percentage will increase from 0% at 100% threat threshold to 100% at 0% threat threshold.

For any given threshold, we also compute the prediction percentage, p_p, which is the percentage of test data events that are included in our threat regions.

If our predictions are random, we expect $p_c \cong p_p$. Good predictions will result in $p_p > p_c$. For example, in a recent experiment, threat regions that cover only 13% of the playbox capture 31% of the threat.

We can report these results in two ways.

1. The statistic $p_p/p_c - 1$ will be greater than zero for good predictions, and approach 0 for random predictions. It is theoretically possible for this value to be less than zero, but in that case there is in fact information in the predictions that is being misused, and could in principle be analyzed to yield an improved prediction. This statistic is a point estimate, valid only for a single threshold.

Figure 3. Threat regions with threshold at 60%.
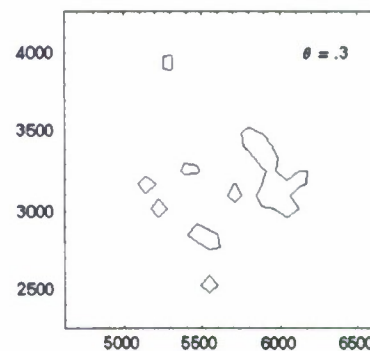
Figure 4. Threat regions with threshold at 30%.

2. We can summarize the performance of a predictor over a range of thresholds by plotting p_p as a function of p_c. Figure 5 shows how we can summarize this graphically. The diagonal line shows the points where $p_c \cong p_p$. Curves **a** and **b** are two different predictors. Curve **a** reflects the preferred predictor, because the statistic $p_p/p_c - 1$ is greater than on curve **b** for every value of p_c. We can compute such a curve by sweeping through the thresholds from 0% to 100%.

It may be helpful to compare this second result with the ROC (Receiver Operating Characteristics) curves often used in analyzing sensors[1]. Such a curve plots the true positive rate (true positives / total positives) against the false positive rate (false positives / total positives). As noted above, we do not possess the data to compute such a curve, but the interpretation of the curve is the same as in Figure 5. A random predictor (or sensor) has a curve close to the diagonal, and the more rapidly the curve rises, the better the predictor/sensor.
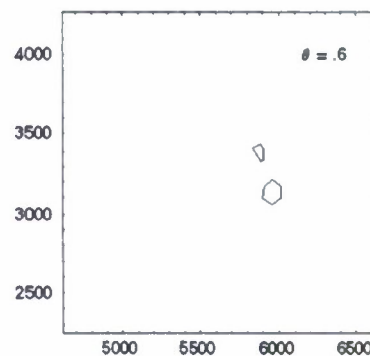
---

[1] J. A. Swets, R. M. Dawes, and J. Monahan. Better Decisions through Science. *Scientific American*, vol. 283, pages 82-87, 2000.

To further quantify the performance of the IED prediction prototype with respect to a Random predictor, we added +/- 2 sigma lines around the Random line. To generate these lines, we modeled our evaluation procedure using a Binomial Distribution. We have N independent repetitions of a simple success-failure experiment where N is the number of future IEDs we are comparing against our predictions. A success is the event that an IED placed at random within the Area of Interest falls inside a threat area. The probability of success under a random trial in this case is the ratio of the sum of the areas of the threat regions to the area of the overall Area of Interest.
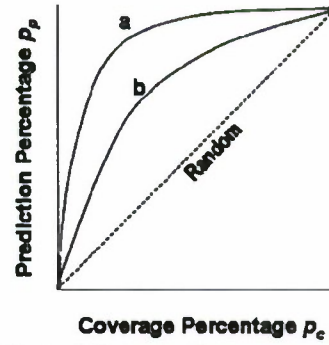

Figure 5. Response Curve.

9.1.2 Evaluation Data

We considered two sets of data. The first set comprised of three synthetic test messages created by Alion (http://www.alionscience.com/) as part of DARPA RAID ARMN evaluation trials. The test messages were

- DRMsg1_9_Past_Events and DRMsg1_8_Future_Events

- DRMsg2_11_Past_Events and DRMsg2_10_Future_Events

- DRMsg3_12_Past_Events and DRMsg3_11_Future_Events

The second set of data comprised 87 actual IED events (obtained through DARPA RAID program) that happened in Baghdad province over a period of few months. The data included date, time, location and zone of the IED event. A zone is defined as city, town or village where the IED event occurred. The data did not contain any information regarding Convoys or patrols. Following were the test cases considered:

Independent of the zone split the test data into two files – All_Zones_Past_Events and All_Zones_Future_Events. Three different pairs of split files were considered

- All_Zones_41_Past_Events and All_Zones_46_Future_Events

- All_Zones_71_Past_Events and All_Zones_16_Future_Events

- All_Zones_79_Past_Events and All_Zones_8_Future_Events

9.1.3 Results

This section includes snap shots of the ROC curves along with a table containing information on Normalized Coverage Ratio for a fixed threshold value. ROC curves display the Random Predictor line along with its +/- 2sigma lines and the scores from running the IED prediction prototype with the baseline synchronic model.

DRMsg1_9_Past_Events input to DRMsg1_8_Future_Events	
Number of Likelihood Regions	100
Likelihood Region Threshold	0.8
Number of Future IEDs	8
IEDs covered by a region	3
IEDs within 50.0m of a region	4
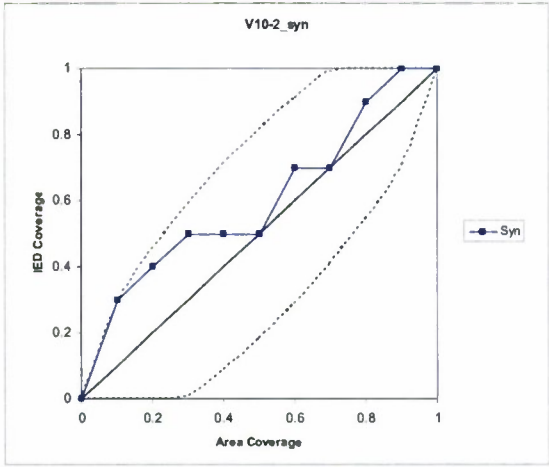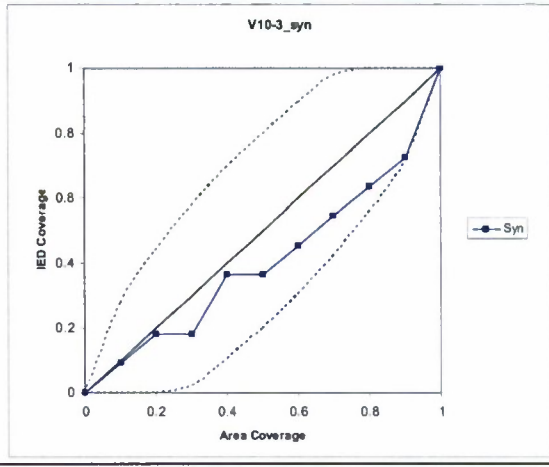
Total Area of Regions	4.171 km^2
Area of AOP	22.794 km^2
Percentage Area Covered	18.3 %
FICP Score	0.375
FICP-A Score	0.306
Random is better	0.0420
Random is worse	0.8331
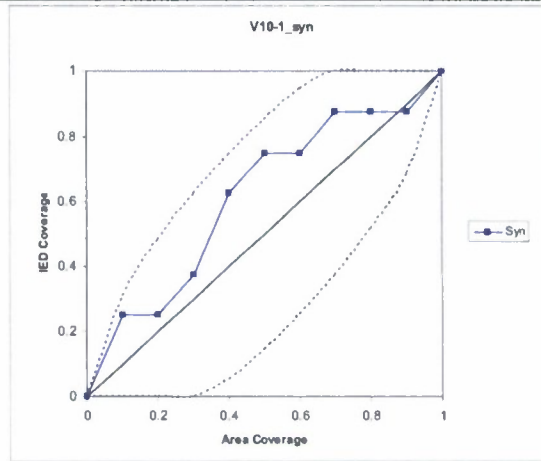
DRMsg2_11_Past_Events and DRMsg2_10_Future_Events	
Number of Likelihood Regions	100
Likelihood Region Threshold	0.8
Number of Future IEDs	10
IEDs covered by a region	1
IEDs within 50.0m of a region	5
Total Area of Regions	3.322 km^2
Area of AOP	20.679 km^2
Percentage Area Covered	16.1 %
FICP Score	0.100
FICP-A Score	0.084
Random is better	0.4942
Random is worse	0.1726

DRMsg3_12_Past_Events and DRMsg3_11_Future_Events

Number of Likelihood Regions	100
Likelihood Region Threshold	0.8
Number of Future IEDs	11
IEDs covered by a region	1
IEDs within 50.0m of a region	4
Total Area of Regions	3.099 km^2
Area of AOP	24.062 km^2
Percentage Area Covered	12.9 %
FICP Score	0.091
FICP-A Score	0.079
Random is better	0.4236
Random is worse	0.2195

All_Zones_41_Past_Events and All_Zones_46_Future_Events

Number of Likelihood Regions	100
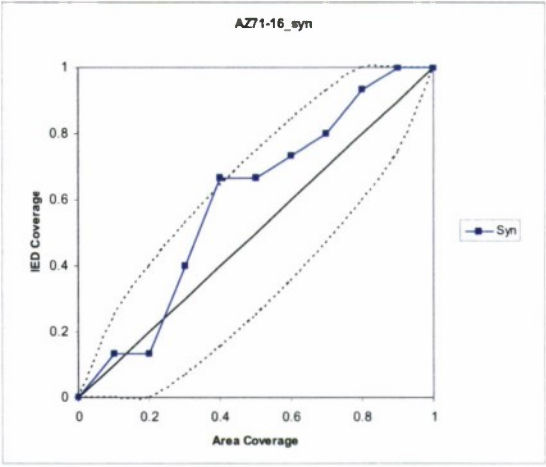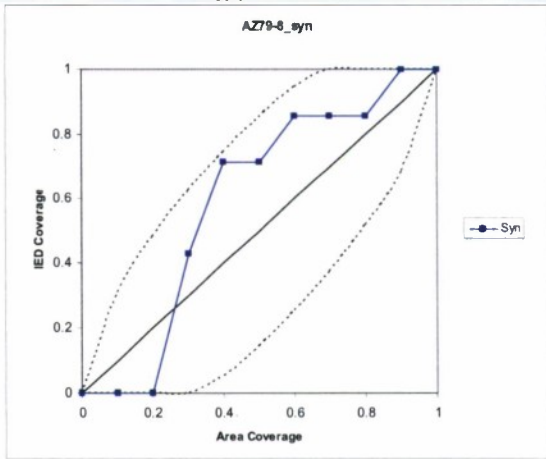
Likelihood Region Threshold	0.8
Number of Future IEDs	46
IEDs covered by a region	5
IEDs within 50.0m of a region	16
Total Area of Regions	0.617 km^2
Area of AOP	11.53 km^2
Percentage Area Covered	5.4 %
FICP Score	0.109
FICP-A Score	0.103
Random is better	0.0351
Random is worse	0.9017
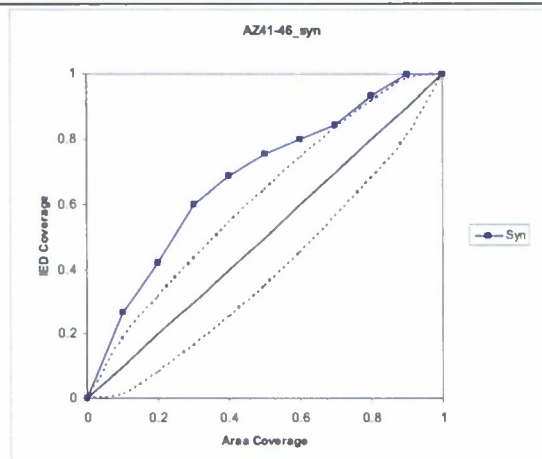
All_Zones_71_Past_Events and All_Zones_16_Future_Events	
Number of Likelihood Regions	100
Likelihood Region Threshold	0.8
Number of Future IEDs	16
IEDs covered by a region	4
IEDs within 50.0m of a region	11
Total Area of Regions	3.03 km^2
Area of AOP	11.53 km^2
Percentage Area Covered	26.3 %
FICP Score	0.250
FICP-A Score	0.184
Random is better	0.4157
Random is worse	0.3606

AZ71-16_syn

All_Zones_79_Past_Events and All_Zones_8_Future_Events

Number of Likelihood Regions	100
Likelihood Region Threshold	0.8
Number of Future IEDs	8
IEDs covered by a region	6
IEDs within 50.0m of a region	6
Total Area of Regions	5.211 km^2
Area of AOP	11.53 km^2
Percentage Area Covered	45.2 %
FICP Score	0.750
FICP-A Score	0.411
Random is better	0.0186
Random is worse	0.9098

AZ79-8_syn

9.2 Enhanced Representations Track

9.2.1 Introduction

A domain for which higher levels of cognition are often considered necessary is coordination in the execution of complex tasks. For example, tests and treatments on a hospital patient can be represented in a treatment plan, which has both internal coordination relationships (some tests must be done in a particular order or within certain time limits) and external coordination relationships between treatment plans (only one MRI machine exists; certain ancillary hospital units prefer to run similar tests in batches to reduce set-up times, etc.) [4]. Another example is the on-line coordination of pre-planned activities in dynamic environments such as military, law-enforcement, or disaster planning scenarios [3]. Several law-enforcement units may wish to surprise suspects at different locations nearly simultaneously so they cannot warn each other. Besides coordinating the surprise itself, some units may require equipment or information whose delivery time is not known in advance. The structure of such tasks can be represented as a graph, specifically, a Hierarchical Task Network or HTN.

All of these types of scenarios have been typically approached by building systems where complex agents have an internal representation of their own plans (and how they relate to the plans of other agents). Examples include CSC agents [6], or unrolling each agent's view of the HTN into a Markov decision process over which MDP techniques can be applied [7], or translating it into a Simple Temporal Network and applying STN techniques [10].

We take a radically different approach. Rather than putting the HTN inside of *complex agents*, we put *swarming polyagents* inside of the HTN. Coordination is achieved, not by conventional inter-agent dialogs based on each agent's individual analysis of the HTN, but by means of interactions among the agents mediated by the structure of the HTN itself. This paper demonstrates this approach by showing how swarming polyagents can operate on an HTN. Specifically, we work with a dialect of the TAEMS task language [5] that emphasizes the importance of resources, both real and virtual, in coordination (thus resource-TAEMS or rTAEMS). What sets this model apart from other (self-organizing) scheduling and execution approaches is that it includes in its reasoning semantic representations of method-execution preferences that require the coordination of multiple entities.

9.2.2 Background

In this section we first summarize the rTAEMS HTN modeling approach, introducing the key terms that define the topology in which the swarming agents operate. Then we briefly introduce our polyagent modeling construct, which uses swarms of simple agents that project specific aspects of the system state into the future for informed decision making.

9.2.2.1 Resource TAEMS (rTAEMS)

A hierarchical task network (HTN) is a collection of events, together with two kinds of relations among them: a hierarchical structure relating tasks to their subtasks, and other relations constraining the order of execution among the tasks. In this paper we focus on the Resource-TAEMS (rTAEMS) dialect of TAEMS as a specific instance of an HTN formalism [1, 5]. For a more detailed introduction and motivation of rTAEMS we refer the reader to [8].
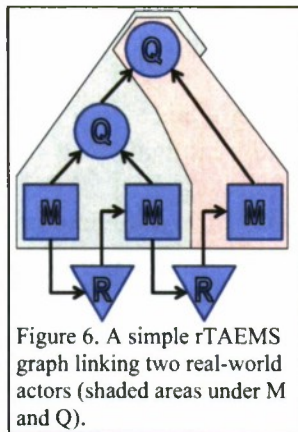
Figure 6. A simple rTAEMS graph linking two real-world actors (shaded areas under M and Q).

Figure 1 shows a simple example of an rTAEMS graph. The circles ("Q") are tasks and subtasks that may be associated with one or many actors in the real world (two shaded areas), and can be subdivided into lower-level activities. Since they also serve as the hosts of the quality accumulation process, we call these rTAEMS nodes **"quality" nodes**. The rectangles ("M") are **"method" nodes**, which are the lowest level of activity. Each method is associated with a single actor and provides a statistical representation of the execution behavior of this activity (e.g., duration, deadlines). Finally, the triangles ("R") are the **"resource" nodes** that are emphasized in the rTAEMS dialect over the traditional TAEMS specification.

The primary purpose of the rTAEMS graph is to coordinate the activities of various actors to *maximize overall quality* achievement while *adhering to any method ordering and timing constraints*. The ordering constraints are imposed by the R nodes in the graph. Those nodes carry a non-negative abstract resource level. Methods that start execution consume a given amount of resources from R nodes that have incoming links to the M node. Methods can only start if the resource levels on the incoming R nodes are sufficient for consumption of the specified amounts. When methods complete, they produce a given amount of resources on R nodes that have incoming links from the M node. The actual amount of resources consumed and produced is defined as static annotations to the R-to-M (consuming) and M-to-R (producing) links. Timing constraints associated with a particular M node may further limit the time window in which the method may be started.

When methods complete, they produce also a given amount of quality for Q nodes that have incoming links from the M node. Similar to R nodes, Q nodes carry a non-negative abstract quality level. In addition, any Q node defines a quality accumulation function (QAF) that combines the quality levels on all incoming links (M and Q nodes) into this node's quality level. That quality level is then used as an input to the QAF at the node's parent, and so on. The current quality achieved by the actors is defined as the current quality level at the root of the Q node hierarchy.

9.2.2.2   *Polyagents Modeling Framework*

For a more detailed introduction to polyagents, we refer the reader to [9]. The "poly" in "polyagent" reflects the fact that each relevant domain entity is represented by multiple agents: a single avatar and multiple ghosts, combining structured self-organizing swarms (ghosts) that explore large search spaces with classical reasoning approaches (optional) in the avatar. Avatars and ghosts differ in four ways.

Multiplicity: Each entity has only one avatar, but may have multiple ghosts existing concurrently.

Scope: An avatar persists as long as the entity it represents. Ghosts are transient. They are continually generated by an avatar at a specified rate, and they die off after a specified period or upon some specified event.

Reasoning: The avatar may use complex symbolic reasoning, and may communicate directly with other avatars. Ghosts are stigmergic, or ant-like. They independently explore alternative paths and coordinate their actions only indirectly, through changes that they make to a shared computational environment. The most common mechanism for ghost interactions with each other and with the avatars of other
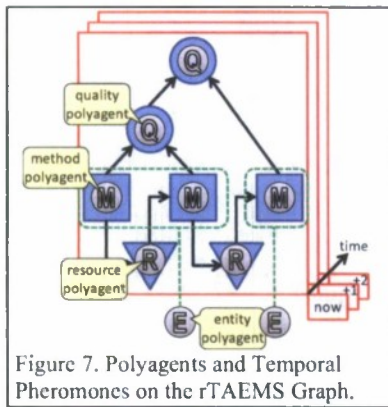
Figure 7. Polyagents and Temporal Pheromones on the rTAEMS Graph.

entities is through digital pheromones, scalar variables that the agents deposit and sense in the environment. As a result, ghost reasoning is a simple and rapid numerical computation over their behavioral model and the pheromone strengths in their vicinity. Traditionally, the topology of the space over which the ghosts swarm is a representation of the geo-spatial aspects of the domain. In this paper, we are demonstrating swarming on HTN graph representations.

Responsibility: The avatar's responsibility is to maintain a model of the domain entity and predict and possibly control its behavior. To that end, it generates and tunes a stream of ghosts, whose mission is to evaluate alternative actions and possible interactions. The emergent result of the ghosts' reasoning can then be used to bias or guide the avatar's actions.

9.2.3   The rTAEMS Polyagents

The various polyagents in our model are coupled through external state variables and temporal pheromone fields that facilitate indirect information exchanges. The pheromone fields are either a probabilistic projection of the state variables into the future (e.g., projected levels at resource nodes or quality nodes), or they encode additional coordinating information required to generate schedules that are correct (enablement) and optimized (quality, deadlines). In our polyagent model, we maintain pheromone fields across the entire graph indexed by a positive temporal offset (future) relative to the current real-world time (avatar time). As ghosts execute their behavioral model, they move through this index of fields from the current time into the future, changing their temporal location.

The manipulation of the pheromone fields by the polyagents' ghosts always follows the same pattern: 1) A ghost carries an internal state that reflects their own estimate of one or more external state variable. In particular, if the external state variable is discrete (e.g., resource level), then the ghost state is discrete as well. 2) The ghost samples pheromone fields at its current temporal location and turns pheromone concentrations into probabilities over state variables. Using their random number generator, the ghost then samples these probabilities to postulate the occurrence of particular events that may change its internal state. 3) Based on these events, the ghost changes its internal state, emulating the change of external variables. 4) Finally, the ghost deposits pheromones at its current temporal location, affecting the event probabilities that other ghosts perceive.

Thus, a ghost emulates a possible evolution of a set of external state variables over time and adds this forecast to the probabilistic representation of the state variable in the temporal pheromone field. This coupling of polyagents through actual or projected state variables allows us to discuss the operation of the polyagent model from the perspective of the information flow among variables first (section 9.2.3.1), before explaining the specific behavior of the individual agents (section 9.2.3.2).

Our polyagent model distinguishes "infrastructure" and "execution" polyagents. The purpose of the infrastructure polyagents is to provide guiding information for the execution polyagents, who in turn construct (ghosts) and execute (avatars) a particular

method schedule. The *infrastructure polyagents* represent individual nodes in the rTAEMS graph and their behavior depends on their node type. Thus we distinguish "resource" polyagents, "quality" polyagents, and "method" polyagents. The *execution*

Figure 8. Entity ghosts consume *R* / produce *S* and *C*.

Figure 9. Resource ghosts consume *S* and *C* / produce *R*.

polyagents' associated with the rTAEMS graph is less localized. They model the behavior of real-world entities that may execute certain methods in the graph. In the current implementation, any M node is associated with one particular "entity" polyagent (Figure 7).

9.2.3.1   *Abstract Information Flows*

We describe the information flows that maintain a correct and optimized schedule for the execution avatars.

9.2.3.1.1   Correct Schedules

The entity ghosts decide when to start and complete a method and accordingly, they deposit temporal "starting" (S) and "completing" (C) pheromones of the selected M node. For correct schedules, the decision whether to start a method depends on the availability of resources consumed by the method. These resource levels are derived from the "resource" (R) pheromone concentrations at the resource nodes linked to the method. Thus, entity ghosts consume R and produce S and C moving through time (Figure 8).

The resource ghosts model the evolution of the level of their R node. As a resource ghost moves through time, it maintains its discrete estimate of the resource level and it deposits this amount of R pheromones. It modifies its estimate by postulating starting and completing events for those methods that consume from or produce to its R node. It postulates these events from the observation of the S and C pheromone fields of those methods. Thus the resource ghosts consume S and C and produce R (Figure 9).

Figure 10 shows that entity ghosts affect the behavior of resource ghosts (by starting and completing methods) while resource ghosts in turn affect the behavior of entity ghosts (by estimating the resulting resource levels). Thus, entity and resource ghosts form a stigmergic feedback loop that results in the emergence of correct schedules where methods are only executed if sufficient resources are available for their consumption.

9.2.3.1.2   Optimized Schedules

The stigmergic interaction of entity and resource ghosts in Figure 10 produces correct schedules that are not optimized according to the quality accumulation defined by the Q nodes of the rTAEMS graph. Also, these schedules do not include optimizations that allow high-value methods with early deadlines to be executed on time.

The quality ghosts estimate the evolution of the quality level at their associated Q node. That level changes when a method's completion adds quality to the node, or when child Q nodes change their levels and change the outcome of the quality accumulation function (QAF).

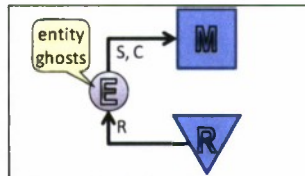Like resource ghosts, quality ghosts observe the S and C
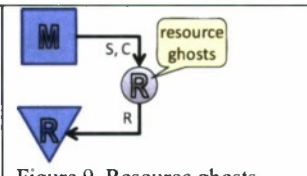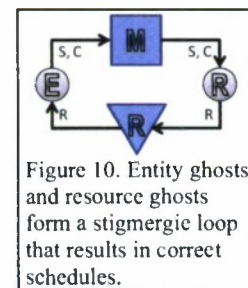

Figure 10. Entity ghosts and resource ghosts form a stigmergic loop that results in correct schedules.

Figure 11. Quality ghosts consume S, C, Q and TQ / produce Q and TQ.
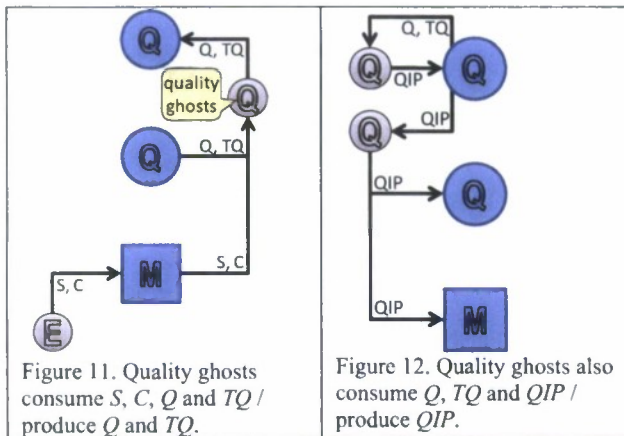
Figure 12. Quality ghosts also consume Q, TQ and QIP / produce QIP.

pheromone levels on those M nodes that provide quality to their node and postulate starting and completing events. Completing events increase the level of quality in the ghost. Quality ghosts also observe the "quality" (Q) pheromone in their node's Q children and estimate their current projected quality level. The estimated quality levels provided by associated M and Q nodes are the QAF inputs. The QAF result becomes the ghost's new quality level and it also determines the amount of Q pheromone that the ghost deposits. Thus, quality ghosts consume S, C, and Q and produce Q pheromones (Figure 11).

The infrastructure polyagents on the Q nodes collectively maintain an estimate of the likely evolution of the quality levels based on the projected execution of methods. To guide the selection of enabled methods, we need to compare the quality of the projected schedule with the total quality that could be achieved. We extend the behavior of the quality ghosts to consider the maximum achievable quality of their M node children and apply the QAFs.

The maximum achievable quality of a method depends on whether the method was already executed or not. If a quality ghost considers a method completed, then the achievable quality is the quality produced. Otherwise, it is the quality that the method is projected to achieve (zero in the case of a missed deadline). The quality ghosts consume S, C and Q to produce "total quality" (TQ) pheromone deposits (Figure 11).

From the calculation of the achieved and total quality profile at the Q nodes, we compute the *quality improvement potential* that remains at the M nodes. This calculation starts at the root of the quality hierarchy, where the quality ghosts deposit a "quality improvement potential" (QIP) pheromone equal to the difference of the Q and TQ values. Quality ghosts on all nodes of the hierarchy (including the root) take their local QIP value and distribute it to their children according to their respective QAF. For instance, in a SUM QAF, the QIP deposits are proportional to the children's TQ contributions. Thus, quality ghosts consume Q and TQ at the root and QIP on all nodes and produce QIP at Q and M nodes (Figure 12).

The concentrations of QIP pheromones on M nodes optimize schedules for high quality. But, QIP alone results in greedy schedules as it does not account for method deadlines. Therefore, method ghosts take the local QIP estimate and combine it with the remaining time to the deadline of their method to compute and deposit the method's "urgency" (U) pheromone.

Finally, we want to induce schedules that execute even low-QIP/late-deadline methods early if they lead to the enablement of high-QIP/early-deadline methods. Thus, we extend the resource ghosts to consume U from their consuming methods and deposit U proportionally to their providing methods if their projected resource level is insufficient to enable their consuming methods.

To produce schedules that are correct in regards to enablement and optimized in regards to quality achievement and deadline adherence, entity ghosts need to consider the resource levels at the enabling resource nodes (R) as well as the urgency levels at the method nodes that belong to their entity polyagent (U). Figure 13 shows the entire information flow among the infrastructure and execution polyagents within the topology of the rTAEMS graph.

There are two ways that coordinating information flows from the future into the decision process of the entity polyagent. Implicitly, the QIP calculation assumes the eventual execution of methods in the total quality estimate. Explicitly, the urgency calculation assesses upcoming deadlines and the propagation of urgency by the method and ghost agents move that measure further upstream.

Figure 13. Information flow among the polyagents in the rTAEMS graph.

9.2.3.2 *Specific Agents*

Now we discuss the operation of the various polyagents in detail. We start with the ghost logic that maintains a schedule forecast for the near future and then describe its execution by the avatars. We present the ghost and avatar operation in sequence, but in reality those two agent types operate in parallel at different time scales (many ghost cycles between any two avatar cycles). First, we discuss how entity and resource ghosts form a correct schedule. Then we include the remaining ghost types to maintain optimized schedules.

9.2.3.2.1 Correct Schedules

Correct schedules emerge in the stigmergic interaction between swarms of entity ghosts and resource ghosts.

9.2.3.2.1.1 *Entity Ghosts*

An entity polyagent represents a particular real-world actor capable of executing a given set of methods. These methods are modeled as M nodes in our rTAEMS graph. The ghosts maintained by the entity avatar establish a correct and optimized schedule in collaboration with ghost swarms from other entity polyagents and supported by the ghosts of the infrastructure polyagents.

Following the general polyagent modeling paradigm, the entity avatar continuously creates entity ghosts at a fixed rate. Upon creation, entity ghosts copy relevant aspects of the current avatar state into their own state and are placed on the temporal location that corresponds to the current real-world time of the avatars. Then, with each ghost decision cycle, the ghost advances one discrete time step into the future until it reaches the model's forecast horizon. There it ceases to exist.

The initial state of an entity ghost comprises the execution history and the current execution state (what, if any, method is being executed now) of its avatar. With each decision cycle, the entity ghost advances this state by choosing to execute methods from its set of M nodes.
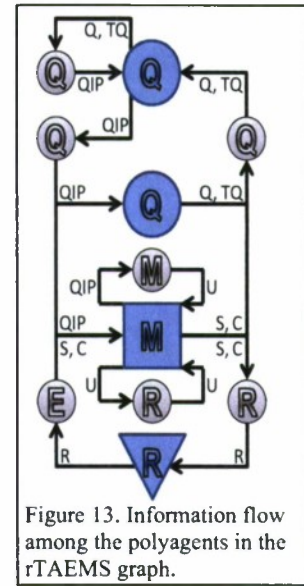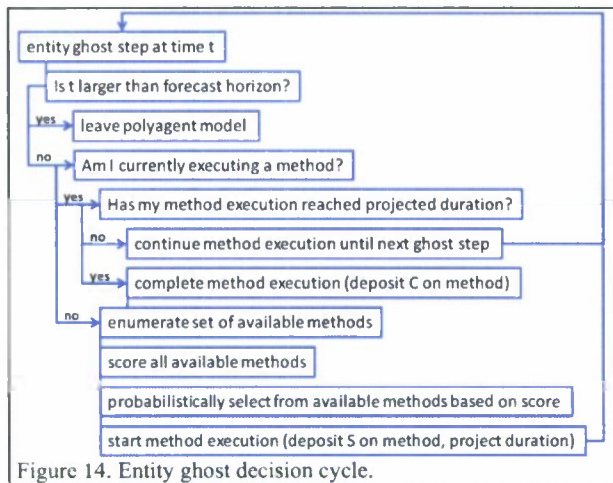
Figure 14. Entity ghost decision cycle.

Figure 14 shows the basic decision cycle for an entity ghost. It first checks whether it has passed the forecast horizon. If not, then the ghost asserts whether it is currently in the process of executing a method. In that case, the ghost needs to decide whether it should consider the method completed or whether it should continue executing the method until its next decision cycle based on an internal duration counter set at the start of the method. If the counter reaches zero, the ghost deposits a unit amount of C pheromone at the method's node in the field indexed with its current time t.

To select a new method, the ghost iterates over all M nodes of its entity and assesses their current availability. The ghost considers a method available, if it has not been executed before by either its avatar or by itself (e.f. re-entrant methods in Future Research 9.2.4.1). Furthermore, the availability of a method also depends on its enablement by R nodes. To determine method enablement, the ghost samples the R pheromone on each providing R node and probabilistically estimates its current resource level. This determination is made under the assumption that the pheromone level is in steady state based on regular deposits by resource ghosts (e.f. [2] for detailed analysis of pheromone dynamics). The method is considered enabled, if the sampled resource levels for all enabling resources are above their respective minimum enablement threshold.

If the resulting set of available methods is empty, the entity ghost just pauses for this decision cycle. Otherwise, it selects a method from that set with uniform probability. Marking the start of the method, the ghost deposits a unit amount of S pheromone on the M node and initializes its method duration counter by sampling the method duration distribution from the node's configuration.

9.2.3.2.1.2   *Resource Ghosts*

We assign a resource polyagent to each R node in the graph. The resource ghosts collectively estimate the evolution of the level of their R node from the current actual level to the model's forecast horizon. This collective estimate is reflected in the R pheromone concentrations on the node, which are sampled by the entity ghosts to decide on a method's enablement state.

A resource ghost carries a discrete resource level. As the ghost is created by its resource avatar, this value is set to the current actual resource level. Also at initialization, the resource ghost determines for each producing or consuming method, whether this method is currently being executed by an entity avatar.

Figure 15 shows the basic decision cycle for a resource ghost, beginning with the check for the model's forecast horizon. The ghost iterates over all providing M nodes. For each such node, it samples S and C pheromones levels, and estimates the probability that this method is starting or completing at this time. Depending on whether the ghost expects the method to start or complete next, the ghost either uses a probability derived from S or C to postulate these starting or completing events. If the ghost postulates a completing event for a producing method, it increments its internal resource level by the amount produced by that method.

After handling possible increments, the resource ghost considers all consuming methods and again postulates starting and completing events. Here the ghost decrements its resource level upon starting events by the amount consumed by the respective method.

It is important to note that the resource level of a particular ghost is permitted to drop to negative values even though the actual resource level of the node never falls below zero. Prohibiting these forbidden states would otherwise artificially constrain the statistical variations that the current method execution patterns may produce.

The ghost completes its decision cycle by depositing R pheromones equal to its new resource level onto its node into the field indexed with the ghost's current time.

9.2.3.2.2   Optimized Schedules

The stigmergic interaction between entity and resource ghosts leads to the emergence of correct schedules. The feedback loop between the execution decisions of the entity ghosts and the resource level estimates produced by the resource ghosts ensures that only those methods are executed that have a high likelihood that sufficient resources for consumption are available. In fact, the entire space of correct schedules is accessible as any correct execution choices may be explored.

Now we show how the infrastructure ghosts process quality accumulation and deadline information to provide additional guidance for the entity ghosts to select optimized schedules from the correct ones.

9.2.3.2.2.1   *Quality Ghosts*

The hierarchy of Q nodes with M nodes at the leaves specifies the accumulation of method-produced quality up to the root of the rTAEMS graph. The quality at the root is the overall performance measure applied to the team of actors whose actions and their interdependencies are modeled. The quality ghosts estimate the evolution of quality levels at each Q node to the forecast horizon and use this estimate to guide the entity ghosts to execution decisions that have the highest potential to improve the root-level quality.

Quality ghosts are very similar to resource ghosts. They carry a level measure for their node (quality instead of resource), and this level is initialized from the current level at their node. If the Q node receives direct contributions from M nodes, the quality ghosts postulate starting and completing events for these methods
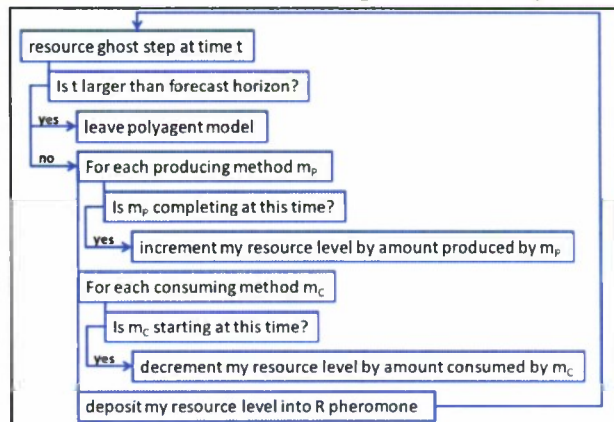

Figure 15. Resource ghost decision cycle.

from the S and C pheromones and increase their internal quality level for any completed providing method.

In contrast to a resource ghost, the internal level of a quality ghost is not just determined by the quality production of associated methods. Instead, the ghost's Q node may also have other Q nodes as children. There the ghost probabilistically derives the currently predicted quality level from their Q pheromones. The ghost uses the estimated method quality contributions and the child Q node levels as input into its QAF. The QAF result determines the current quality level of the ghost and the ghost deposits Q pheromones of this amount into its Q node.

As discussed in section 9.2.3.1.2, quality ghosts also estimate the accumulation of total quality that can be achieved at their respective node. The mechanism for the creation of TQ pheromone fields is very similar to the Q pheromone generation. For TQ the quality ghost tracks the execution of the providing methods through the S and C pheromones but postulates total quality production as long as the method has not passed its deadline without being completed. Using its node's QAF, the quality ghost combines achievable quality of its providing methods with probabilistically determined TQ levels of any child Q node. It deposits the resulting total quality level as TQ pheromones on its node.

With the Q and TQ fields established by the quality ghosts, we now have sufficient information about the possible improvement of the overall root-level quality. The quality ghosts *of the root node* deposit the quality QIP pheromone as the current difference between their Q and TQ levels. All temporal QIP fields *below the root node* are maintained by the quality ghosts of the parent node. These ghosts sample the QIP pheromone at their own node and distribute that value as QIP deposits to their children according to the local QAF. For instance, if the QAF at the parent is a SUM or a MAX, then the parent's QIP is distributed to the children proportionally to the difference between Q and TQ at the respective child. Thus child nodes that still offer the largest gain in quality get assigned the largest quality improvement potential. Other QAFs, like for instance a MIN, trigger an inverse proportional distribution of the parent's QIP.

The distribution of parent QIP to child nodes of a Q node does not distinguish between Q node and M node children, distributing the root QIP to down individual methods. Thus, the combined operation of all quality ghosts maintains a quality improvement potential profile starting at the current avatar time out to the forecast horizon, identifying which methods (if enabled and executed) may provide the largest gains for root-level quality.

9.2.3.2.2.2 Method Ghosts

In scenarios without deadlines, QIP alone would be sufficient to guide the entity ghosts towards an optimal schedule. In this case, entity ghosts may just greedily pick methods that offer the largest QIP and fill in any remaining smaller quality gains later. But if these low-gain methods are associated with a deadline, then they should be executed earlier than high-gain methods with later deadlines. Therefore it is necessary to combine the QIP information of a method with any deadline associated with the method to determine the urgency with which the method should be selected if it is enabled.

The method ghosts perform this simple calculation. Their internal state accumulates the likelihood that their method has been started at or before their current ghost time. If the entity avatar already started the method, then the ghost's starting

Figure 16. Method ghost decision cycle.

Figure 17. Urgency "propagation" by resource ghosts.
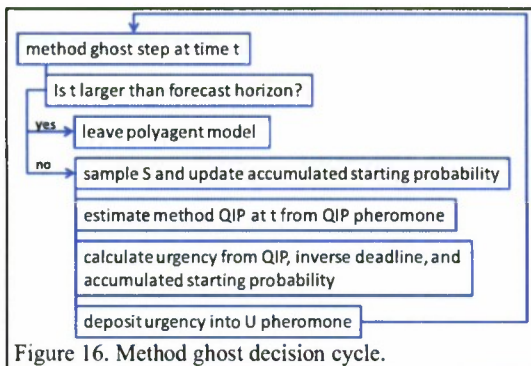
estimate is 100% right from its initialization. Otherwise it accumulates starting probabilities sampled from the S pheromone field in each method ghost step.
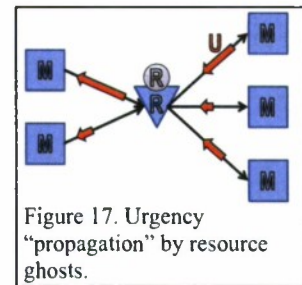
In each decision cycle (Figure 16) the method ghost samples its local QIP level and divides it by the time that remains until the deadline or the forecast horizon (whichever comes first). It then multiplies this value with 1 - accumulated starting probability – the likelihood that the method is not yet started. The resulting urgency measure grows with increasing quality improvement potential, with an approaching deadline, and with decreasing starting probability. The method ghosts deposit this urgency value into the U pheromone field.

9.2.3.2.2.3   Resource Ghosts

One final step in the urgency calculation is necessary to ensure that methods with high indigenous urgency (close to deadline, high QIP, low starting probability) get enabled by upstream methods in time even if these predecessor methods themselves have low indigenous urgency. In effect, we want to selectively "propagate" urgency upstream along the method enablement relationships expressed by the R nodes between them.

We extend the behavior of the resource ghosts beyond our initial description. After completing the operations associated with the creation of correct schedules, a resource ghost sums up the urgency of consuming methods that currently have insufficient enablement by their providing resources. Each such urgency value is weighted with the resource level that the respective method would consume from the ghost's resource. The resulting "resource urgency" value is then distributed (U pheromone deposits) proportionally among all providing methods according to the level of resource they would be contributing. The temporal index of these deposits is offset by the duration of these methods, increasing the urgency to start these methods in time.

9.2.3.2.2.4   Entity Ghosts

In the previous sections we showed how the ghosts of the infrastructure polyagents create a rich information environment across space (rTAEMS graph) and time (up to the forecast horizon) based on global quality improvement potential and method deadlines resulting in localized urgency fields at the method nodes. Now we discuss how the entity ghosts take this information into account when making their execution decisions.

The summary of the entity ghost decision cycle in Figure 14 already includes the necessary steps: "score all available methods" and "probabilistically select method based on score". To create correct schedules it was sufficient to select among the available methods randomly. With method urgency information available, the entity ghost scores its available methods by their U pheromone concentrations. Thus, methods with higher urgency have a higher likelihood of being executed.

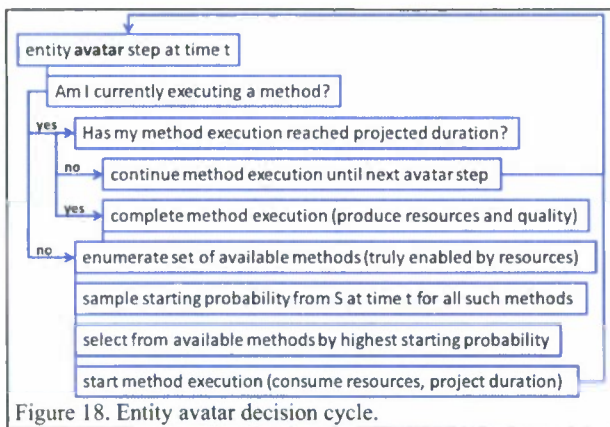| entity **avatar** step at time t |
| Am I currently executing a method? |
| yes → Has my method execution reached projected duration? |
| no → continue method execution until next avatar step |
| yes → complete method execution (produce resources and quality) |
| no → enumerate set of available methods (truly enabled by resources) |
| sample starting probability from S at time t for all such methods |
| select from available methods by highest starting probability |
| start method execution (consume resources, project duration) |

Figure 18. Entity avatar decision cycle.

Methods that have an urgency of zero will not be executed since they neither contribute additional quality themselves nor enable other methods that may provide quality. Thus it is possible that even though methods are enabled for a particular entity ghost, the ghost may still decide not to execute anything. This is a desirable behavior as it allows entity polyagents to ignore unproductive methods.

9.2.3.2.3 Executing Schedules

We had mentioned before, that in the general polyagent model, the avatar may be the host of complex (cognitive) reasoning process about its entity. For the rTAEMS model presented in this paper, such complex reasoning is not necessary, because already the infrastructure and execution ghosts collectively maintain a correct and optimized schedule in the distribution of S and C pheromones on method nodes. Thus, all necessary reasoning about which (if any) methods should be actually executed next by the entity avatars is performed by the swarming ghosts of the system. All an **entity avatar** has to do now in its decision logic is to exploit the guidance that is generated by the exploration of the information landscape by its ghosts.

The entity avatar (Figure 18) executes similar decision logic as an entity ghost (Figure 14) in regards to its overall execution behavior. If, in a particular decision step, it is already executing a method, it decreases its method duration counter and completes the method if the counter reaches zero. If the avatar is ready to select a new method for execution, establishes a set of available methods. It is up to the entity avatar to ensure that its execution remains correct in regards to *actual* enablement and deadlines of its methods. Its ghosts used R pheromones to estimate enablement and though it is unlikely that this estimate is wrong at the beginning of the forecast window (close to the actual system state), entity avatars still have to enumerate their set of available methods based on actual resource levels at the enabling nodes and exclude those methods that it actually executed before or that have run out of time.

From the set of available methods, the entity avatar selects the method that has the highest starting likelihood (from the S pheromone) at the first ghost cycle. If there is more than one such method, the avatar has no further guidance and selects among the maximum likelihood methods randomly. If the highest starting probability is still below a configurable threshold, then the avatar does not select any method for execution and pauses instead until its next cycle.

In the forecasting component of the polyagent model, entity ghosts do not consume or produce resources directly. Instead, resource ghosts observe the starting and completing probabilities of their associated methods and maintain the resource-level forecast in the R pheromone. Conversely, the execution by the entity avatars constitutes the real and irreversible start and completion of methods. Thus, as an avatar starts a method, it actually consumes resources (decrements resource levels), and when it completes a method, it actually produces resources (increments resource levels) and

quality (increments quality levels). As a consequence, the decision process of **resource avatars** is empty.

While the resource avatars are impoverished in their behavior, **quality avatars** still play an important role. They have to track the actual production of quality by the entity avatars and recursively roll those up to the root node of the quality hierarchy. This roll-up provides an immediate picture of the currently achieved quality by the polyagent system.

Finally, **method avatars** simply register the execution of their method by an entity avatar to establish the correct initial state of their ghosts.

9.2.4 Evaluation Experiments

We report on experiments with our rTAEMS polyagent system. First we discuss capability experiments on artificially constructed graphs that highlight particular challenges. Then we report on benchmark tests against traditional TAEMS approaches.

9.2.4.1 Capability Experiments

Below is a small sample of the various experiments that we performed to test agent interaction, fine tune agent parameters, and analyze the scalability of the algorithm by increasing the number of avatars and methods.

9.2.4.1.1 Stepped Deadline Graph (Single Entity Avatar)

The graph contains ten methods M_{1-10} with a common duration of one, producing a quality of one each (no quality preferences). Method M_i has a deadline at $t=i$ (stepped deadlines). The optimal sequence of method execution by an avatar is highly constrained by the deadlines. If any one method is executed out of sequence, a loss of quality is observed.

Without quality preferences, the entity ghosts provide sufficient guidance for the avatars to execute the stepped graph flawlessly (see screenshot in Figure 19). But we find that differences in quality production may lead to greedy, out-of sequence execution of higher-value methods. A tuning parameter that balances the impact of deadlines with the preferences expresses by quality production suppresses this greedy behavior to a point. But in the case of the stepped graph, the modeler who constructed the fully deadline-constrained graph should not have added any conflicting quality preferences.

9.2.4.1.2 Quality/Deadline balance Graph (4 Entity Avatars)

As observed in the previous section, higher quality offered by one method may overwhelm the urgency to execute another to meet its (or its dependants) deadline. This series of experiments demonstrates the existence of a balance point where quality greed overwhelms deadline constraints.

We specify a graph with 4 avatars {A, B, C, D} and their respective two methods {M[A-D]$_1$, M[A-D]$_2$}. Methods M[A-D]$_1$ are all assigned a deadline of 1, while methods M[A-D]$_2$ are assigned a deadline of 2. The quality produced by the method was set as follows: M[A-D]$_1$= 1, MA$_2$=1, MB$_2$=2, MC$_2$=3, and MD$_2$=4. The optimal sequence
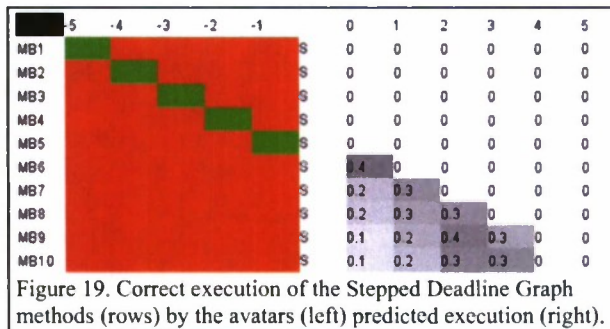
Figure 19. Correct execution of the Stepped Deadline Graph methods (rows) by the avatars (left) predicted execution (right).

Table 1. cTAEMS Benchmark Characteristics and Results.

Characteristics			
Name	#Entities	#Methods	#Nodes
MayOptNLE4	6	12	28
MayOptContingent3	3	18	43
Results			
Name	Optimal Quality	Mean Quality	Std. Dev. Quality
MayOptNLE4	65	65	0
MayOptContingent3	66.5	66.5	0

of method execution for any avatar is M_1 before M_2. Without the aforementioned (9.2.4.1.1) tuning parameter, the system is able to execute this optimal sequence only for avatar A. The other avatars were led by their ghosts to execute their second method ($M[B-D]_2$) first because of the quality improvement potential they were offering. Thus, the deadlines of $M[B-D]_1$ were reached and their quality contributions lost.

The tuning parameter affects the contribution of QIP to the urgency of a method and thus balances the competing optimization goals of maximizing quality and meeting deadlines. Repeating the experiment with a decreased QIP impact, avatar B now also executes MB_1 before MB_2, but $M[C-D]_1$ still expire.

These experiments highlight that the designer of rTAEMS graphs must not only adhere to the correct syntax of the graph, but must also be aware of the emergent dynamics of the polyagent system that resolves competing optimization goals.

9.2.4.1.3 Stepped Deadline Graph (Scaling Tests)

To explore the scalability of our polyagent approach, we first increase the number of stepped methods (see 9.2.4.1.1) executed by a single entity avatar to 500 (M_{1-500}, $dM_i=i$) and then increase the number of avatars that are associated with the stepped methods to ten (alternating avatar-method association $A_1=\{M_1, M_{11}, M_{21}, ...\}$, $A_2=\{M_2, M_{12}, M_{22}, ...\}$). In all cases, our polyagents were able to produce the optimal (stepped) execution sequence, naturally with increasing computational cost (linear with #methods, #avatars). It is worthwhile to point out that all polyagent interactions in our model are local on the rTAEMS graph. Thus, the distribution of this system over many computational hosts for a distributed group of coordinating entities is very straight forward.

9.2.4.2 Benchmark Experiments

In order to compare our polyagents' performance with related work, we experimented with two cTAEMS task networks taken from the May, 2007 evaluation trials of the DARPA COORDINATORS program (K. Decker, personal communication). The task networks used in these trials were intended to be simple enough to be solvable by an optimal cTAEMS algorithm, while still being complicated enough to evaluate and compare the performance of non-optimal, heuristic solvers.

The two networks chosen were "MayOptNLE4" and "MayOptContingent3" (Table 1). We converted the original cTAEMS networks into equivalent rTAEMS networks, preserving semantics and resulting quality. Accounting for the probabilistic nature of our approach, we executed 25 replications of each network with a different random seed. The results in Table 1 show that our approach achieved optimal results.

9.2.5 Conclusion and Outlook

Typically, swarming and even polyagent applications place their agents in shared computational environments with geographic topologies. These metric and reasonably continuous spaces provide the agents with sufficient space to explore alternative trajectories with minor variations where the non-linear dynamics of the agent system amplifies these variations when they offer improvements to the system's performance.

We demonstrated here how swarming agents may be deployed on the non-metric and discontinuous topology of a process graph, using the metric and continuous temporal domain and the distribution of numeric resource and quality levels as the source for those minor variations that are essential to the adaptiveness of self-organizing algorithms.

We align our research with traditional Artificial Intelligence approaches and focus on Hierarchical Task Network (HTN) descriptions of the constraints and preferences in the execution of abstract methods by a group of entities. In particular, we adapt the TAEMS representation for HTNs to place a greater emphasis on the mediation of method-execution through shared resources and collectively achieved quality (stigmergic coordination). On the rTAEMS graph representation of methods that are enabled by the availability of resources produced by other methods and whose execution produces quality that is aggregated to a system-level quality achievement, we place "infrastructure" polyagents on each node that project the evolution of the state of their node forward in time. The entities that are using the rTAEMS graph to coordinate their activity are also represented by polyagents, driving through their projected and actual execution of methods the evolution of the infrastructure agents.

We discussed in detail the population-level dynamics of the complex polyagent system on rTAEMS, specified the decision logic of the agents that make up the swarming component of the polyagents ("ghosts"), and reported on capability and benchmark experiments. We were able to show that our polyagent approach to scheduling and execution of HTNs is capable of achieving optimal performance while offering the ability to dynamically reschedule to adapt to changing environments and to distribute the process among multiple hosts associated with the coordinating entities. (Due to the stochastic nature of the algorithm, optimal performance cannot be guaranteed on more complicated problems.)

The ONR CIED STIFLE project has come to an end. But in a related project, we are now extending the rTAEMS polyagent system in several directions. We are expanding the applicability of the rTAEMS process model by supporting the execution of a method more than once (re-entrant methods) and potentially by different entities (shared methods). Also, to model opposing "sides" among the entities for instance in war-games, we allow rTAEMS graphs to have more than one quality root.

The main extension comes from the specialization of the methods. In the rTAEMS version reported in this paper, a method is characterized by abstract attributes such as its duration or deadline and we assume that the method always concludes successfully (producing quality). The specialized method nodes will include a detailed execution model that simulates the execution of the method in a geo-spatial model. From that simulation, we derive dynamically the expected duration of that method and the amount of quality it produces. Thus, methods could have varying duration and success depending on the spatial context in which they are executed.

9.3   Model Analysis Track

9.3.1   Prediction Horizon

An important aspect of our polyagent framework is the explicit reasoning about possible future states of the system performed collectively by the ghost populations of all polyagents. Each polyagent maintains a set of ghosts that each emulates the avatar's behavior from a set point in the recent past (hind-cast horizon) to a given point in the near

future (forecast horizon). A ghost's emulation of the avatar's behavior typically involves the probabilistic interpretation of a sub-symbolic behavioral model within a range of "personality" parameters. In some applications, we also apply evolutionary learning to the ghosts' personality parameters against observed entity behavior.

As ghosts emulate the behavior of their avatar from the present point onwards into the future, they evaluate the likelihood and possible outcome of interactions of their avatar with other avatars in the model. This evaluation is based on state-likelihood information communicated among the ghosts through spatio-temporal pheromones. The combination of the probabilistic behavioral model and the indirect interaction of ghosts through a shared environment establishes a positive feedback loop that drives the model to convergence on a small set of the most likely future trajectories of the model, establishing a prediction of the future. This prediction is constantly refined and updated as new information enters the system.

As with all efforts to predict the future behavior of a non-linear complex system, our ability to look into the future is limited by the rapid divergence of possible system trajectories even under very similar initial conditions – popular known as the "butterfly effect." Therefore, our limited computational resources (computational cycles available to the ghosts combined with the complexity and uncertainty of the current situation determine an effective

Figure 20. Effective limit of polyagent prediction capability.

prediction horizon beyond which the trajectories of the individual ghosts cannot be combined into a meaningful prediction of likely future states.
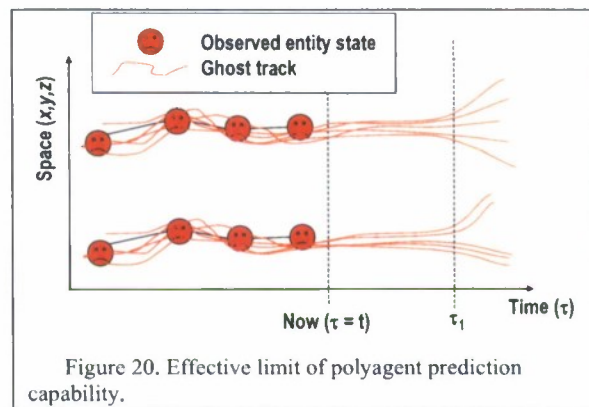
Figure 20 illustrates the existence of an effective prediction horizon in our polyagent model. We hypothesize that an analysis of the emergent ghost trajectories may enable a polyagent to manage the available ghost processing cycles, avoiding effort being invested in spatio-temporal regions of the model that are beyond our analytic capability.

We took a first step towards such an auto-adaptive mechanism by analyzing a simplistic model of predictive control and demonstrating the impact of the actual prediction horizon on the emerging system performance. We published a paper to the 2007 International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'07), describing this model and our experiments and conclusions (see Publications Table).

9.3.2 Quantum-Mechanical Analogy

We explored the empirical analogy between polyagent systems (one distinct entity following a swarm of probabilistic behavioral emulators interacting through fields) and a quantum mechanical interpretation of the world (distinct particles following a classical path but being the dual to waves that may interfere with other waves). In particular we have been studying the classical two-slit experiment that exposes the quantum-mechanical particle-wave duality and we started to explore the notions of interference and frustration driven by external circumstances and internal preferences.

In our polyagent models, we use multiple ghosts of an entity to create probability fields which in turn guide the movement of the single avatar of the entity. Thus, abstractly, our polyagent models resemble the particle/wave duality known from Quantum Mechanics in physics. In STIFLE, we are exploring the possibility that this resemblance may actually yield formal tools or approaches from the physics domain that improve the design or performance of our polyagent models.

We started our exploration of the Quantum Mechanical Analogy by exploring the classical two-slit experiment that demonstrates the particle/wave duality of electrons. In this

Figure 21. The classical two-slit experiment.

experiment, a source for electrons is placed on one side of a gate with a screen that shows the resulting electron distribution on the other side. If the gate has only one slit, then the resulting distribution of the electrons on the screen is centered around the center of the direct particle path. But if the gate has two narrow slits, then the resulting distribution shows the effect of interference of the electron as a wave passing through both slits (see **Figure 21**).
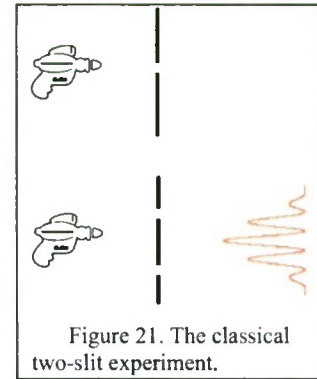
To replicate the experiment in the polyagent framework, we instantiated a simple model with one avatar at the location of the electron source, emitting ghosts with a random heading and speed (constrained to parameterized intervals). We also extended the framework to include obstacles (walls) at which ghosts are either absorbed or reflected. Thus, ghosts behave like replications of an electron particle, forming peaks along the classical path behind the slit(s). Figure 22

Figure 22. Model of the two-slit experiment.

shows a snapshot from the execution of the model in which a stream of ghosts (black dots) is "filtered" by a two-slit gate.
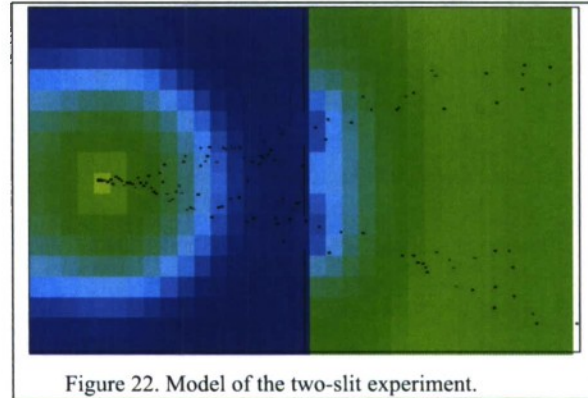
To replicate the wave interpretation of the electron that leads to the interference pattern behind the two slits, we implemented a new infrastructure similar to our application-independent pheromone infrastructure. While the pheromone infrastructure supports spatial aggregation, diffusion, and evaporation of information in an approximation of chemical pheromone markers in social insect colonies, the "Quantum Wave" infrastructure will support diffusion and interaction (interference) of information approximating wave dynamics.

In many applications we have been using the pheromone infrastructure to offload computational requirements typically associated with truth maintenance and team coordination from individual agents to the shared environment, thus simplifying the agent code. We believe that the Quantum Wave infrastructure may have unique information processing capabilities complementing the pheromone infrastructure that would support other agent tasks.

The snapshot in Figure 22 shows an example of the Quantum Wave infrastructure dynamics as the avatar modulates the local "field displacement" with a sinc function. The

local displacement is propagated by the infrastructure dynamics and, as it reaches the back of the two-slit gate, leads to interference reminiscent of the interference in the two-slit electron experiment.

Just like the pheromone infrastructure, the Quantum Wave infrastructure is composed of multiple cells that are locally linked into a graph structure. In our experiment in Figure 22, the graph forms a rectangular grid that is only disrupted at the location of the walls of the gate. Each cell in the grid may have a displacement along one or more independent dimensions, the equivalent to the different flavors in the pheromone infrastructure. Agents may sense or modulate this displacement.

The dynamics of the infrastructure determine how the displacement of one cell influences the displacement of its neighbors. Many different update rules for the field displacement are possible. We have experimented with a few. For instance, the averaging rule specifies that the displacement of a cell at time t+1 should be equal to the average displacement of its neighbors at time t. Under such a regime, the amplitude of the

Figure 23. Displacement by distance to avatar in linear chain of Quantum Wave cells.

displacement of any cell in a finite graph would eventually approximate a constant displacement modulated by one agent onto one cell. Or, if the displacement by the agent changes periodically, then the amplitude of displacement of other cells in a regular grid decreases proportionally with the distance from the agent (see Figure 23).

The averaging rule approximates the dynamics of energy transfer in a set of connected heat storage bins. We have also experimented with an update rule where each cell is the equivalent of a pendulum, continually transferring potential to kinetic energy and back. In this case, we are able to approximate the actual propagation of wave fronts through the graph without energy loss (similar to electro-magnetic waves).

9.3.3 Two-Bridge Problem

In the following, we present a detailed discussion and analysis of a simple model ("the two-bridge problem") that follows the topology of the two-slit experiments but created frustration and interference effects in a more agent-like fashion. While the two-bridge problem primarily exposes the analogy to quantum-mechanical systems and their analysis, it does reflect, in an abstract sense, spatial decisions similar to those that need to be made while emplanting an IED.

This section will describe the simple two-bridge problem and to study the ground states of the system. The primary purpose is pedagogical. We will show how to frame a class of problems we are interested in, so that dynamics in both physical space and decision space are appropriately expressed. We will also show how the ground states of the "Hamiltonian" can result in qualitatively different outcomes depending on the values of the parameters, and how frustration can lead to effects which (superficially at least) mimic some aspects of quantum systems.

Consider a river which can be crossed by one of two bridges (Figure 24). There are 2 agents (either people or platoons), represented by squares, who are positioned, one
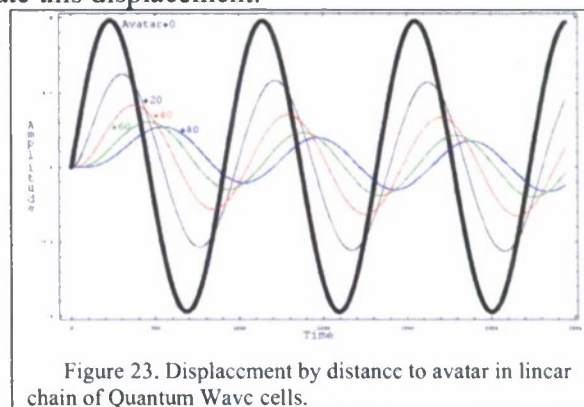
at each of the two bridges as shown below. There are three possible sites to bivouac after crossing the bridge, represented by circles.

Associate with each of the two agents a "spin", S_i (i=1,2). Each spin can take on one of 3 values corresponding to the bivouac site chosen by that agent. It is convenient to represent each spin as a complex phase, $e^{i\phi_i}$ so that the three choices correspond to three different values of ϕ, 0, $2\pi/3$ and $-2\pi/3$. Let sites (1,2,3) corresponds to

Figure 24. Topology of the Two-Bridge Model.

$$\phi = (\ 2\pi/3,\ -2\pi/3 \text{ and } 0), \text{ respectively.}$$

There may be different circumstances that dictate degrees of preference for the agents to bivouac at different locations. For example, agent 1, may be extraordinarily tired, and so may have a strong preference to bivouac at site 1. On the other hand, there may be better accommodations and supplies at site 3, so that agents will prefer to travel the extra distance to site 3. Also, there may be some reason why the agents prefer to bivouac together, regardless of the site.

The effect of such preferences on the choice of bivouac site can be expressed by forming a Hamiltonian (or utility function). Minimization of the Hamiltonian amounts to finding the statistical ground state of the system. Under some circumstances the actual choice of bivouac site will minimize the Hamiltonian. In any case, analyzing the grounds states is a first step in understanding likely scenarios for bivouac choice and their sensitivity to preferences.

We describe this system with the following Hamiltonian:

$$A = -(JS_1 \cdot S_2 + H_1 \cdot S_1 + H_2 \cdot S_2) + h.c.$$

Here J is a scalar, and the $H_i = h_i \exp(i\theta_i)$ are complex numbers. The H_i express preferences of each of the agents to choose a given site. For example, if $\theta_1 = 2\pi/3$ then agent 1 will prefer to bivouac at site 1, and the strength of that preference will increase the larger h_i is. The values of θ_i do not have to be limited to $2\pi/3$, $-2\pi/3$ or 0. Suppose, for example that agent 1 has an equal preference to be either at site 1 or 3, but prefers not to be at site 2. This can be accommodated by choosing $\theta_1 = 2\pi/6$. By choosing different values of θ_1, we can accommodate varying relative preferences among the 3 sites. J expresses the degree of preference that the two agents bivouac together. If J is large and positive, there will be a strong preference for the two agents to bivouac in the same place. If J is negative, the agents will want to bivouac in different places.

To analyze the ground state of A, it is convenient to rewrite it in the following form:

$$A = -[J \cos(\phi_1 - \phi_2) + h_1 \cos(\vartheta_1 - \phi_1) + h_2 \cos(\vartheta_2 - \phi_2)]$$

To simplify our problem, we assume symmetry in preferences between the agents. That is, we consider only cases in which $h_1 = h_2 = h$ and in which $\theta_1 = -\theta_2 = 2\pi/3 - \delta$. Then we have

$$A = -[J \cos(\phi_1 - \phi_2) + h \cos(2\pi/3 - \delta - \phi_1) + h \cos(-2\pi/3 + \delta - \phi_2)]$$

We now can study the minima of A for different values of J, h and δ. Here we will only outline the qualitative behavior of the system for different ranges of the variables.
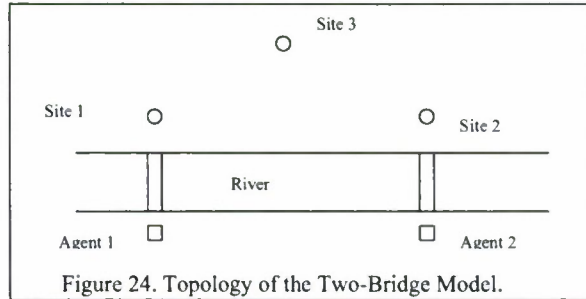
9.3.3.1   δ = 0

    This is the ease in whieh, if there is any preference of the agents for a site, it is for the site closest to the bridge that they cross (assuming that h>0). In this case, if h>>J, then the agents will prefer to bivouae at the site near their respeetive bridges. We eall this the elassical ease. If J>>h, it will be most important to the agents that they bivouae together, and, if h is nonzero, then A will be minimi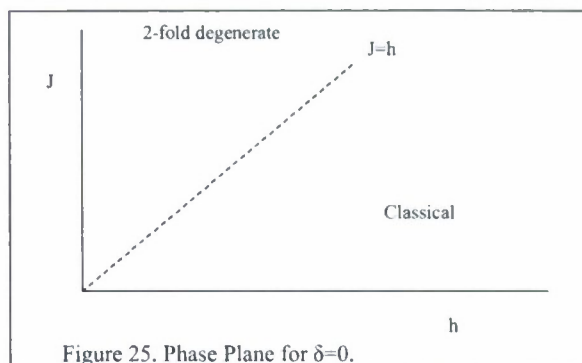zed if the agents bivouae at either site 1 or site 2. At this level of the analysis, there is nothing to distinguish between these two choices. The ground state is said to be 2-fold degenerate. The cross-over between these two behaviors oeeurs when J=h. For J>h the ground state has the two agents together at one site, while for J<h, the ground state corresponds to eaeh agent forming a bivouae at the site nearest his bridge. This behavior is summarized in the graph in Figure 25.

Figure 25. Phase Plane for δ=0.

9.3.3.2   0 < δ ≤ π/6

    This qualitative behavior continues for small values of δ, although the separatrix between the elassieal and 2-fold degenerate domains changes, taking on a gradually larger slope. (This may seem counterintuitive, since a small increase in δ means that an agent has a less strong preference to bivouae near his bridge than if δ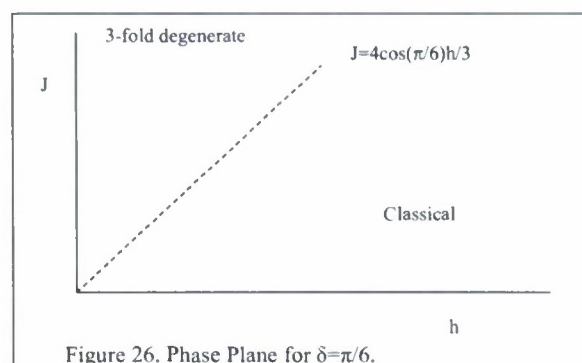=0. However, when δ>0, there is a greater eost in having the distant agent bivouae near the opposite bridge. That is, $[\cos(4\pi/3) - \cos(4\pi/3-\delta)]$ is greater than $[1-\cos(\delta)]$ for small δ.) Figure 25 continues to describe, qualitatively, the ground state behaviors for δ<π/6. When δ=π/6, the state with both agents choosing site 3 becomes degenerate with the choices of sites 1 or 2, and so the upper region of Figure 25 now becomes 3 state degenerate. That is, for δ=π/6 and J sufficiently large, relative to h, both agents will bivouae at the same site, and will have no preferenee of the site. Note that this is the first appearance of the "quantum mechanical" solution, reminiscent of the two-slit diffraction experiments. The separatrix in this ease is the line J=[4hcos(π/6)]/3 or, roughly, J=(1.1547)h. (Compare with the case δ=0, in whieh the separatrix is given by J=h.) This is summarized in Figure 26.

Figure 26. Phase Plane for δ=π/6.

9.3.3.3   π/6 < δ ≤ π/3

    For δ>π/6, and above the separatrix, the solution with both agents at site 3 has a lower value of A than the solution with both agents at sites 1 or 2. So this region now becomes purely quantum mechanical. Qualitatively, for larger values of δ, this region continues to be dominated by the quantum meehanical solution. The reason is that larger

δ expresses a greater preference by the agents for site 3 than for the site closer to their bridge. The slope of the separatrix is a maximum at $\delta=\pi/6$ and is approximately equal to 1.1547. As δ increases above $\pi/6$, the slope of the separatrix decreases. Thus, at $\delta=\pi/6$ two important things happen. Above the separatrix, the quantum mechanical ground state dominates the 2 fold degenerate ground state, and the slope of the separtrix decreases. This is fairly interesting and probably suggests that this should be thought of as a process of jumping from one sheet which represents the two-fold degenerate ground state, to another which represents the quantum mechanical solution as we pass through $\delta=\pi/6$. When $\delta=\pi/3$ the slope of the separatrix is zero and the entire positive quadrant of the (J,h) plane is dominated by the quantum mechanical solution. This makes sense: as δ increases, each agent (independent of the question of whether the agents bivouac together or not) increasingly prefers to bivouac at site 3, rather than at the site closest to its bridge. When $\delta=\pi/3$, the angular separation between site 3 and the maximum of the cosine (i.e., the value of, say, ϕ_1 such that $2\pi/3-\delta-\phi_1=0$) is equal to the corresponding angular separation between this value of ϕ_1 (ϕ_2) and site 1 (2). Thus, the h terms in the energy are neutral with respect to the agents' choice between its home site (closest to its bridge) and site 3. A positive value of J only reinforces this behavior, since that term expresses the preference to bivouac together. Thus, the unique preferred solution when $\delta=\pi/3$, for (J>0, h>0), is the quantum mechanical one.

9.3.3.4 $\pi/3 < \delta < 2\pi/3$

Note, finally, that as δ increases beyond $\pi/3$, the slope of the separatrix becomes negative with the quantum mechanical solution dominating above the separatrix, even if J<0, as shown in Figure 27. (Note also that for this range of δ, the quantum mechanical solution is always preferred over the 2 state degenerate solution.) This can be understood by realizing that for

Figure 27. Phase Plane for $\pi/3>\delta>\pi/6$.

$2\pi/3>\delta>\pi/3$ each agent prefers to bivouac at site 3 rather than at its home site. On the other hand, when J<0, there is a tendency for the agents to prefer to bivouac separately. But if h is large enough for a give J<0, the preference to bivouac at site 3 can overcome the interagent antipathy and the quantum mechanical solution will be the solution of choice.

9.3.3.5   *Remarks*

1. The modeling lesson here is that we can generate models that incorporate both the physical (by which term we include sociological) constraints of a system and the constraints implicit in the decision space.

2. This analysis is based on a comparison of the ground state energies associated with different solutions. There are at least 2 ways in which the ground state energies can fail to provide good estimates of the solutions.
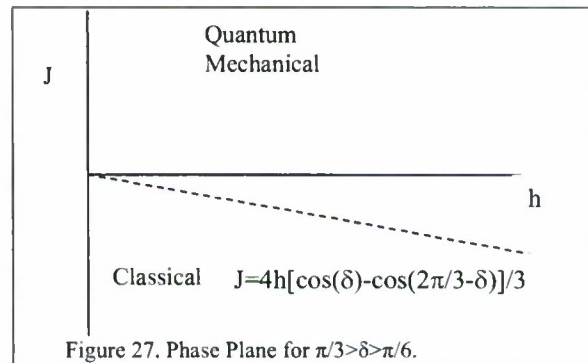
A. The ground state analysis can also be understood as maximizing a global utility function. Maximizing local utility functions may or may not lead to different states. This is similar to the difference between "physical" equilibria and Nash equilibria.

B. The ground state analysis minimizes the global energy. In the face of uncertainty (e.g. if there are too many microscopic variables to follow) one may want to consider minimizing the free energy rather than the energy.

3. It might be interesting to redo this problem using poly-agents. The use of poly-agents may bear a closer formal resemblance to a kind of path integral approach, and so may exhibit the diffraction analogy more clearly. If so, then it may be possible to relate that analogy (which is itself formally similar to the quantum mechanical diffraction system) to the analysis of the ground state presented here. This latter incorporates a formalism in which frustration can be expressed in a straight-forward way. Therefore, by comparing the poly-agent approach with the approach presented here, we may gain a better handle on the relationship between frustration and the emergence of solutions that have quantum-like characteristics.

9.3.4   Theoretical Analysis

After an initial broad exploration of possible aspects of polyagent models that may be amenable to formal analysis, we settled on a promising subset and started to identify promising techniques and approaches. We continued our investigation of the effect the polyagents' prediction horizon has on the performance of the system, and we started to develop an extension of the pheromone model to encode aspects of the predicted agent state in addition to information about spatial presence in pheromone fields.

9.3.4.1   Rationalizing the Research Agenda

The motivation behind pursuing the "Model Analysis" track in the STIFLE project is that we need robust, formal underpinnings to engineer predictive polyagent models reliably and to extract useful insights from them. In our models, we assign a polyagent to a domain entity. The domain entity may be known (with full or partial information), or may be hypothesized (e.g., number and location of specific IEDs during production). Each **polyagent** comprises two kinds of software agents - a **single avatar** which manages the record of the known entity history and potentially a single predicted future, and a **population of ghosts** that perform distributed probabilistic reasoning about the past (e.g., model fitting) and possible futures (e.g., behavioral extrapolation) on behalf of the avatar. Ghosts and avatars use digital pheromone fields to build up knowledge (learning) and to exchange information among each other (communication). We are studying the underlying structure of the agent and field aspects of our unique modeling construct.

9.3.4.1.1   Agents Exploring and Exploiting Multiple Futures

In predictive polyagent models, avatars issue a stream of ghosts that sample multiple futures for their associated entity. On the one hand, the multiplicity of these futures is derived from possible variations of internal parameters of the ghosts' behavioral models (e.g., What would my future look like if I behaved like this...?). On the other hand,

probabilistic estimates of the outcome of interactions with the environment of the entity, including other entities, may result in alternative futures.

In STIFLE's analysis track, we seek to understand in detail the relationship between the multitude of future paths explored by the ghosts in a polyagent and the actual trajectory that the avatar derives from the ghosts' feedback. Producing a formal understanding of this relationship, we allow us to determine the appropriate number of ghosts that need to be generated to produce a sufficient sample of future trajectories, thereby ensuring statistical significance while avoiding excessive computation (wasted processing cycles).

Furthermore, based on an understanding of the ghost-avatar path relationship, we will be able to select the most appropriate mechanism for the avatars to exploit the information gathered by their ghosts along different trajectories. For instance, under different circumstances the avatars might consider just the aggregated pheromone fields, analyze detailed ghost trajectories, or let ghosts compete and then follow the most successful one. The structure of the relationship between an avatar and its ghosts strongly suggests the possible applicability of a least-action formulation of the problem. We are exploring this and other formal methods to address these questions.

9.3.4.1.2 Field-Based Reasoning Mechanisms

Ghosts manipulate digital pheromone fields that are distributed over a given topology to emulate entity interactions between ghosts of different avatars, communicate performance estimates among ghosts of the same avatar, or guide their respective avatars in their decision processes. Our analysis track seeks to determine criteria for the selection of the most appropriate set of fundamental dynamics that govern the pheromone fields. For instance, to what extent should pheromone fields partake of heat-like (diffusion) dynamics or wave-like dynamics, in which there is the possibility of interference?

The selection of the appropriate field dynamics in a particular polyagent model is important for several reasons:

- We need to maintain structures and patterns within the pheromone fields that provide useful information to the agents.

- We need to develop mechanisms that avoid "muddying the waters" as more pheromones are deposited by the agents. To this end, it may be appropriate to develop more refined cancellation mechanisms for information carried by the pheromones than just evaporation (time-based cancellation).

- We need to develop a better understanding of pheromone aggregation mechanisms. In particular, pheromone fields may not be additive. If the same event that is encoded in a pheromone deposit occurs twice, the pheromone aggregation should not necessarily just be doubled, since the occurrence of two events may indicate an even greater probability.

One possible approach to these challenges with which we are experimenting is the use of complex-valued pheromone fields to support more complex interactions.

9.3.4.1.3 Agents vs. Fields - Applying the Appropriate Reasoning Paradigm

Our algorithms link the behavior of avatars and the pheromone fields which they generate and sense (with the help of their ghosts). The emergent dynamics of this relationship inevitably lead to a distribution of information about the system across the

two aspects. We know from experience that for some reasoning processes (e.g., discrete representation of intentions or goals), an entity-based model centered around the software agent of the avatar is more tractable and effective. For others (e.g., probabilistic emulation of engagements), we have found it more effective to evolve probability fields (encoded as digital pheromones) through successive ghost populations that perform Monte-Carlo samples of alternative possible behaviors.

Our analysis track seeks a formal understanding of these dynamics and the relations between the two aspects that they imply, so as to construct, control, and analyze our models in a more principled way.

- In *constructing* a model, these insights will guide us in deciding (for example) whether ghosts should simply report alternative independent futures to their avatar (using agent-to-agent messages) or feed information back to one another (through their pheromone fields).

- In *controlling* a model, these insights will (for example) enable the system itself to decide in real time when an avatar has learned all that it can from the fields its ghosts have generated, so that it should take a discrete action as an entity and begin the cycle of ghost exploration anew (or invoke alternative reasoning mechanisms).

- In *analyzing* the output of a model, these insights will guide us to the aspect of the system (entities vs. fields) most likely to contain the information of interest to answer a particular question that we pose.

9.3.4.1.4   Large-Scale System Dynamics

Whether represented by agents, fields, or both, our predictive polyagent models include a large number of active real-world entities interacting in and with a complex geo-spatial and sometimes cultural environment which changes dynamically over time. Such segments of the real world are often rife with complex constraints and (implicit) utility functions that easily result in frustration of one or more entity preferences at any given time. The potential impact of frustration on the emergent system-level dynamics can be observed in much simpler systems (for example, spin-glass systems).

Our analysis track seeks to develop a formal understanding of the emergence and evolution of frustration in the underlying decision processes of polyagent systems. Such an understanding will support the analysis of our predictive model. In particular, it will help explaining some of the large-scale effects we see in our predictions. Furthermore, it will improve our decision support function, as we can then develop execution strategies for over-constrained systems, where agents face multiple inherently incompatible objectives, to propose useful actions in the face of high levels of noise or limited, partial information.

9.3.4.1.5   Diffusion Models and Prediction Horizons

Prediction is at the heart of polyagent models. Consequently, a central question in polyagent and related models is the question of the best prediction horizon. As we predict farther into the future, we open up a broader range of strategic possibilities. On the other hand, we expect that far future predictions are generically less reliable. One might suppose that predictions at some intermediate value carry the best combination of quantity and reliability of information. What is the structure of information gleaned at

various prediction horizons, and can we deduce general principles that will guide us to the best choice.

Previously, we had analyzed a polyagent model of "Cowards and Rambos" engaged in a complex pursuit game on a toroidal arena. As we reported in our publication at the AAMAS'07 conference (see "Publications" table below), we were able to observe the effect of the extent of the Cowards' prediction horizon on their ability to avoid being chased down by the Rambos. Qualitatively, the Cowards perform very badly with a short horizon, then quickly gain significant improvements as their horizons expand, only to lose those gains gradually, as we kept increasing the horizon even further. While our extensive simulations of this model showed the existence of a "sweet spot" in the polyagents' prediction horizon, the complexity of the model prevented us from completing any formal analysis of the main drivers for these dynamics or rules for estimating the parameters that put the model at the "sweet spot" for a given configuration. Therefore, in this period of performance, we moved to a simpler model, which nonetheless captures the essential dynamics.

We start out with the most simple polyagent model possible. The model has one polyagent on a 2D landscape, frozen in real time (no avatar decisions needed), issuing a continuous stream of randomly walking ghosts. Each ghost executes a fixed number of decision steps until it reaches the polyagent's prediction horizon. Starting at the location of the avatar, in each step the ghost moves a fixed distance in a randomly selected direction. It also deposits a fixed amount of pheromone at a node of a rectangular lattice (Pheromone Infrastructure Place) that covers its current location on the continuous landscape. The deposit is tagged with the current offset of the ghost's simulated time relative to the origin of the avatar's time. Therefore, two ghosts that arrive at the same spatial location but after a different number of decision steps will deposit pheromones into different "time buckets" and will not be able to sense or influence each other.

We implemented this model and executed the simulation long enough that a sufficient number of ghosts were able to complete their run from the present time (frozen avatar time) to the future prediction horizon. At the end of the simulation, we extract the resulting pheromone concentrations at all Pheromone Infrastructure Places and for all time offsets visited by the ghosts. From that data, we extract spatial pheromone fields for each time offset.

Because of the specific dynamics of the Pheromone Infrastructure that we chose for this experiment (fixed evaporation process, propagation of pheromones disabled), we find that the concentrations across a field for a given time offset are proportional to the probability that a randomly moving avatar would be found inside a given Place after the number of steps indicated by the time offset of the field.
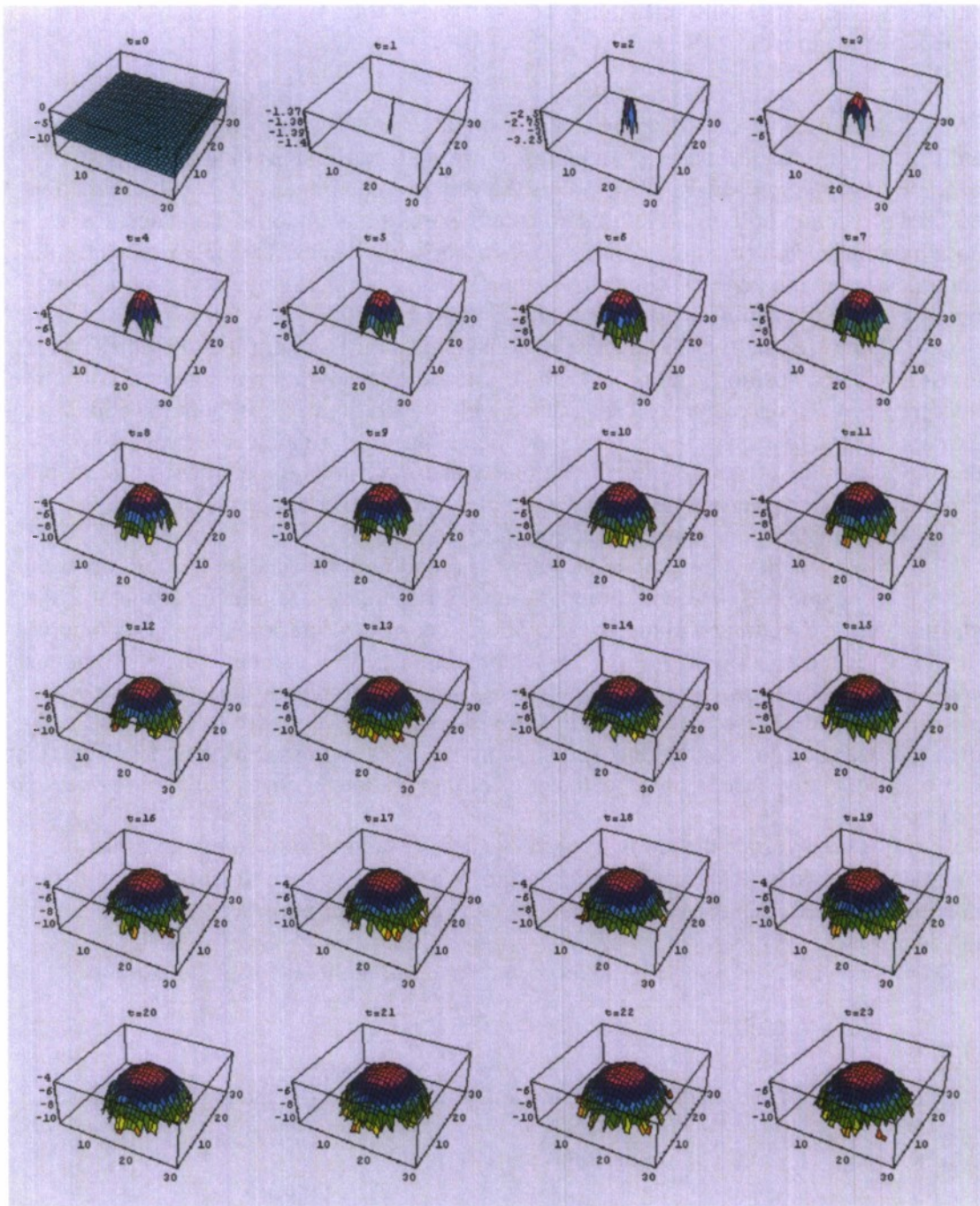
Figure 28. Logarithm of the pheromone concentrations across a spatial field with fixed time offset.

Figure 28 shows plots of the various spatial pheromone fields out to a time offset of 23 steps. We normalized each field such that the sum of all concentrations adds up to one and each normalized concentration represents the probability to encounter a randomly walking avatar at this space-time coordinate. For our graphics, we plot the logarithm of this probability across the space visited by the ghosts. Not surprisingly, as we go more steps into the future, our ability to pinpoint a randomly walking avatar diminishes (the field spreads wider). But the shape of the logarithmic plot immediately

exposes the similarity with diffusion processes, which leads us to the following formal analysis of the model.

9.3.4.1.5.1 *Discussion of the Formal Analysis*

The model described above can be thought of as a random walk approximation to a diffusion process. Although a simple random walk (or diffusion) model is excruciatingly simple, our analysis points the way to some interesting conclusions about optimal prediction horizons, the fundamental dynamics that underlie them, and some possible unexpected emergent consequences of those dynamics. The only potential complication in our analysis comes from the existence of pheromone evaporation, but will show below that this is not an essential complication

Pheromone density can almost be considered a measure of the probability that a ghost is at some position, r, at time, t. The existence of evaporation mitigates this interpretation. To make the correspondence with probability of ghost position complete, we should have no evaporation. We would then simply renormalize the pheromone density so that the integral of the pheromone density was always one at each time step. This renormalized pheromone density without evaporation could be taken as the probability of ghost position as a function of time.

There is a close correspondence between random walks and diffusion processes that we will exploit. Before continuing, though, it is important to remember that it is the ghosts that are executing a random walk. Therefore, strictly speaking, we should apply the diffusion equation to the ghosts, not the pheromones. In this model, the pheromones don't diffuse. However, because the ghosts lay down pheromones at each time step, the pheromone field is a trace of their routes (for the moment, we ignore evaporation), and therefore we can consider that the diffusion process applies to the pheromones. Note, that this may not be the case in other realizations of the relationship between pheromones and ghosts.

It is well known that a random walk process can be described by the diffusion equation. Here we will sketch the relationship. We will not derive the diffusion equation in detail. There are many extant derivations of it, and for our purposes we don't need all the details.

In any case, ignoring for the moment evaporation, the diffusion equation is

$$\frac{\partial u}{\partial t} = D\nabla^2 u$$

Where u is the pheromone density and $\nabla^2 = \sum_{i=1}^{d} \frac{\partial^2 u}{\partial x_i^2}$ is the gradient operator in

d dimensions. The solution to this equation is

$$u(\vec{r},t) = \frac{1}{(4\pi Dt)^{d/2}} \exp\left(-r^2/4Dt\right)$$

where r^2 is understood to be the vector magnitude of the position.

The scale for diffusion is set by the value of D, the diffusion parameter. It is worthwhile seeing, at least roughly, how that is related to a random walk on a lattice. It is straightforward to show (see, for example, F. Reif, *Fundamentals of Statistical and Thermal Physics* (McGraw-Hill, 1965)) that the diffusion constant is, roughly, related to

a random walk process in which molecules suffer random collisions by $D \approx \frac{1}{d}\bar{v}l$,

where $\bar{v}$ is a measure of the average velocity of the random walker, and l is the distance between collisions. In our case, assuming that the lattice spacing is set to one, and the ghost time step is defined as one, $D \approx 1/2$, in two dimensions.

Estimate for the prediction horizon: What do we want for a good prediction horizon? Suppose we have avatar A seeking to predict the position of avatar B. Suppose further that the ghosts of avatar A can sense the pheromones laid down by the ghosts of avatar B. Then, the prediction horizon, T, will be reasonable if two conditions are met. 1. Avatar A must have a reasonable probability of having ghosts in the region where avatar B may be, at time T in the future, and 2. the pheromone field of avatar B (or of it's ghosts) at time T in the future should have some significant structure so that the ghosts of avatar A can sense privileged regions or directions associated with the (future) position of avatar B. From the solution, it is clear that for a fixed time, dependence of the probability distribution is governed by the exponent. If $r^2/4Dt \gg 1$, there is only a small probability of finding a ghost (or substantial pheromone field density associated with those ghosts). If $r^2/4Dt \ll 1$ there is a good chance of finding a ghost, and, the pheromone field, while varying with distance, does not vary dramatically. When $r^2/4Dt$ is on the order of 1, there is both a significant chance of finding a ghost, as well as fairly rapid variation in the pheromone field. Now, return again to the problem of ghosts from avatar A wanting to predict the position of avatar B. Suppose that the ghosts from both avatars are driven by the same (random walk) dynamics and lay down pheromones in the same way. Suppose that the typical inter-avatar distance at some time τ is S. Then, if we choose a prediction horizon, T, such that $S^2/4DT \sim O(1)$, we will satisfy both conditions for a good prediction horizon. At a distance of about S (or S/2) from avatar A, there will be a reasonable probability of finding ghosts from avatar A. At a distance of about S (or S/2) from avatar B the pheromone field from the ghosts of avatar B will have a reasonable spatial gradient allowing the ghosts from avatar A to make reasonable directional judgments about the position of avatar A. In fact, it can easily be shown that the radial gradient of the pheromone field is maximal when $r_{max}^2=2Dt$. Setting $t=T=S^2/4D$, we have $r_{max}^2=2DT=S^2/2$, so that, the maximum gradient occurs in the region which the ghosts will typically sample (i.e. the average inter-avatar distance), given our estimate for the optimal prediction horizon.. Intuitively, this is easy to see. This value of T is about the time when the cloud of ghosts (and their attendant pheromone fields) from the two avatars begin to touch each other without excessive overlap.

The addition of evaporation: From the point of view of the diffusion equation, we can add evaporation by modifying the diffusion equation as follows:

$$\frac{\partial u}{\partial t} = D\nabla^2 u - \rho u$$

where ρ controls the evaporation rate. Either by inspection, direct substitution or by separation of variables, it is straightforward to see that the solution to the diffusion equation is easily modified to take the evaporation term into account. The result is

$$u(\vec{r},t) = \frac{1}{(4\pi Dt)^{d/2}}\exp(-r^2/4Dt)\exp(-\rho t)$$

The effect of evaporation in this model is only a multiplicative term that depends only on t. In particular, the spatial variation of u at a fixed time is not affected by evaporation. Therefore, evaporation won't change the estimate of the optimal prediction horizon. Evaporation only affects the relative scale of the size of u at different times, and effect that does not enter into our estimate of prediction horizon.

Speculations on the emergent structure of prediction horizons: Based on our considerations of the random walk problem above, and our preliminary results on the Rambo-Coward game, we expect that a general feature of prediction in polyagent and related systems is the existence of a small range of best prediction horizons. In general, predictions over short horizons provide little information, while predictions over long horizons are greatly degraded and noisy. We saw this effect explicitly in the Rambo-Coward game, and some thought shows that the same pattern will emerge in the simple random walk model above. The question is, how generic is this behavior and what is its nature. One possibility, which we are exploring, is that the "sweet spot" in a game with nontrivial dynamics, will lead to emergent behavior reminiscent of the Minority Game. In that game, the role of prediction horizon is played by a variable that carries with it the amount of information agents use to make decisions. As a function of that variable, there is a phase transition which occurs at the value of the information which is optimal for system performance. The two phases separated by this transition in the Minority Game have characteristics reminiscent of the very short and very long prediction horizons in the Rambo-Coward and random walk models.

9.3.4.1.6   Phased Pheromone Fields

Current implementations of digital pheromones represent each flavor as a scalar. One consequence of this convention is that while pheromone aggregation is local (at the point of deposit), attenuation can only be done globally (by evaporation, which affects all places in the pheromone landscape indiscriminately). This asymmetry leads us to inquire whether we can design a local attenuation mechanism, analogous to interference in a wave system. This might be achieved by representing pheromones as complex numbers, with both amplitude and phase.

Interfering pheromones could be useful in representing certain decision situations, in which the presence of multiple options leads to an outcome different from what would emerge in the presence of only a single option. Sometimes consideration of multiple options leads a human reasoner to conceive of a new option that combines features of previously articulated options, in effect making a decision that is "in between" the earlier options in the decision space.

The implementation of a phased pheromone is straightforward. It simply consists of two scalar pheromones, which are interpreted as the real and complex components of the phased pheromone. The vector addition rules for complex numbers mean that aggregating and evaporating the real and complex components individually results in the correct behavior for the pair. The innovation is in requiring the agents to deposit and sense the pair as a pair.

We explored the operation of complex-valued pheromones in the context of a military example. Consider a dispersed company of marines who are moving eastward. They need to cross a river, and after making the crossing, they will want to bivouac together. Subject to these constraints, they want to make progress eastward. We wish to implement this coordination without direct communication among the marines.

If there is only one bridge, they will naturally rendezvous at a bivouac due east of the bridge. However, if there are two bridges, rapid movement of the force suggests that the marines should divide the load between them, and in this case the best bivouac is not due east of either bridge, but at a point midway between them. The coordination mechanism should a) guide the marine toward one or the other bridge, dividing the marines between the bridges in roughly equal numbers, and then b) lead them back to a bivouac midway between the bridges.

The first coordination problem, dividing the marines between the two bridges, is straightforward and requires only scalar pheromones. It uses the standard ant routing algorithm, in which the ghosts deposit "home" pheromone as they move away from the avatar, then once they have found the target, deposit "target" pheromone as they climb the home pheromone gradient back to the avatar. Under this algorithm, when the avatar is far from the river and equidistant from the bridges, ghosts will initially form paths across both bridges, but stochastic effects will break the symmetry between the paths. One path will be slightly stronger than the other, and as the avatar draws closer to one of the bridges, its ghosts will tend to cross the closer bridge, reinforcing the path over that bridge and leaving the avatar to that choice. Since the stochastic effects vary from one agent to the next, the avatars will be distributed over the two bridges. The proportion of distribution will depend on the environment. If the environment does not favor one bridge over the other, the avatars will be roughly equally divided. If there is an environmental constraint (say, heavy foliage in front of one bridge that slows the passage of ghosts), more avatars will follow the easier route and fewer will select the more difficult one.

To solve the second coordination problem (converging on a bivouac midway between the bridges), we use a phased pheromone. The phase of the pheromone that is deposited is initially 0 when the ghost leaves the avatar. It then shifts either to $2\pi/3$ after the ghost traverses the north bridge or $4\pi/3$ after it traverses the south bridge. Ghosts modulate their deposit of pheromone based on the phase of the pheromone already deposited at the location (defined by the vector sum of the deposits so far). Ghosts deposit more pheromone the closer the phase is to π. Thus ghosts who encounter pheromone deposited mainly by other ghosts who crossed the same bridge that they crossed will deposit less pheromone than ghosts who encounter pheromone deposited by ghosts from both bridges, resulting in a pheromone peak between the two bridges on the east side of the river. In turn, the avatar follows the gradient of the magnitude of this pheromone to find its way to the peak.

9.3.5 Walker Models

To further extend our model analysis, we focused on the analysis of simple polyagent systems that, in their basic nature, share common mechanisms with those used in the IED prediction prototype. Specifically, we looked at populations of avatars that seek to cluster or homogenize in a given space. We use similar mechanisms of attraction and repulsion to integrate the motivational model of the insurgents in our prototype with the actions of Blue and the constraints of the geo-cultural landscape.

We developed and analyzed two related models. The "Probabilistic Walker Model" deploys ghosts to establish and sample predictive pheromone fields around the avatars, who then use the feedback from the ghosts to move closer to (clustering) or away

from (homogenizing) other avatars. The "Mean-Field Walker Model" abstracts away from the variability introduced by finite populations of ghosts and has the avatars make their movement decisions based on estimated fields that would be established and sampled "in the mean" by an infinite number of ghosts.

9.3.5.1   *The Probabilistic Walker Model*

In these experiments, two avatars start out separated by a distance, S, and try to move closer to one another. At the start of execution, both avatars send out 15 ghosts. The ghosts walk randomly, depositing pheromone and sensing the other avatar's pheromone. After N steps, the ghosts who see pheromone from the other avatar at their current location report back to their avatar. The avatar sums the vectors from itself to the reporting ghosts, and takes a step in the resulting direction. If no ghosts report back, or if the vector sum is zero, the avatar does not move. This cycle is

Figure 29. Avatar steps vs. ghost steps per avatar

repeated until the two avatars meet. The number of cycles until the avatar meets is recorded and plotted versus N, the number of ghost steps per cycle.

The number of cycles it takes the avatars to meet (to attain their goal) is the measure of performance of this system. The number of steps the ghosts take each cycle represents the prediction horizon, or amount of "look ahead" the avatars use in their decision process. The result expected is that there is an optimum prediction horizon. If the ghosts take too few steps, they are unlikely to randomly stumble across pheromone from the other avatar's ghosts, providing little information to the avatar, causing many
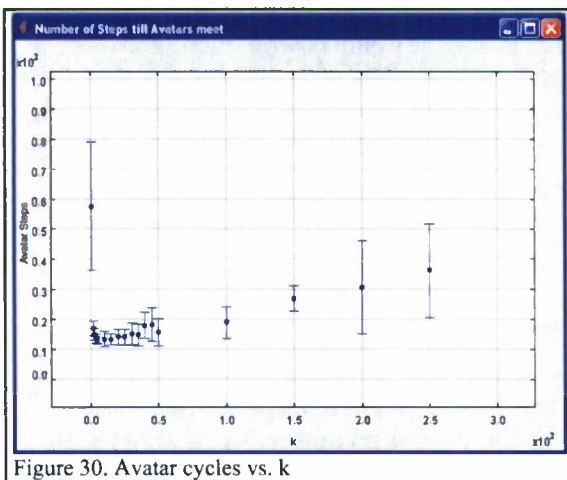

Figure 30. Avatar cycles vs. k

cycles to pass before the avatars will meet. If the ghosts take too many steps, the field of possible ghost interaction expands and overlaps, so the avatars are more likely to take steps in an erroneous direction, thus causing an increase in the number of cycles. Figure 29 shows the results of this experiment with N ranging from 8 to 100. Each run with a different N was repeated 10 times. The mean value for number of cycles is plotted with error bars indicating +/- one standard deviation. The plot shows 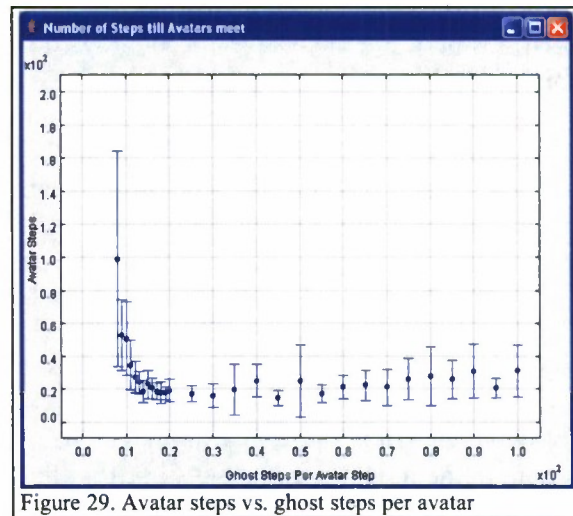the expected results: poor performance with N set below 10, maximal performance in the middle range, and degraded performance as N gets larger. In the region as N gets larger, the mean value of number of cycles stays relatively flat, but the variance over multiple runs increases, which indicates that poorer performance is more likely than in the middle range.

In these experiments, as the avatars get closer to one another there is an increasing overlap of the sensing ranges of the avatars, thereby effectively increasing the number of ghost steps per cycle. To eliminate this effect, the experiment was modified. The number of ghost steps per cycle was modulated by the inter-avatar distance. The number of steps at each cycle was set to $k*S^2/8$, where k is now the independent variable. The experiment was run with k varied from 1 to 250. Figure 30 shows the results of this experiment with number of cycles plotted versus the modulation factor, k. This shows a result similar to above, but with the performance more clearly degraded as k runs out into the hundreds.

9.3.5.2 *The Mean-Field Walker Model*

As we found in our simulation experiments discussed in the previous section, the emerging dynamics of the Probabilistic Walker Model depend on how far the ghosts run out into the "future" (length of random walk). What we did not explore experimentally, is the dependence on the number of ghosts sent out per avatar-decision-step. It is intuitively clear that if there are only a few ghosts, then the pheromone fields and "sensing" events that guide the avatars are very dependent on the random trajectory of individual ghosts rather than the aggregated behavior of the ghost population. To remove this effect from the analysis of the prediction horizon, we considered the following mean-field abstraction of the Probabilistic Walker Model.

9.3.5.2.1 Abstraction 1: Circular Pheromone Fields

The ghosts of an avatar perform an N-step random walk starting from the current location of the avatar on the map. In each step, the ghost deposits a fixed amount of a pheromone identifying the avatar. In the unlikely event that all N steps of a ghost are headed into the same direction (straight line) the ghost would end up at a distance of N*stepLength away from the avatar. Of course, most ghosts will end up less than this maximum distance away from the avatar.

If we assume that the avatar releases an infinite number of ghosts, then we can assume that their N-step random walks will result in non-zero pheromone concentrations in a circular field of radius N*stepLength. Thus, for a "mean-field" approximation of the Probabilistic Walker Model, we can assume that ghosts would sense pheromones from another avatar up to a fixed radius away from the avatar. Note that at this point we are not making any assumption about the concentration of the pheromone in this circular field.

Conclusion: We assume the presence of pheromone concentrations at a fixed radius around an avatar without using ghosts to generate this circular "announcement" field.

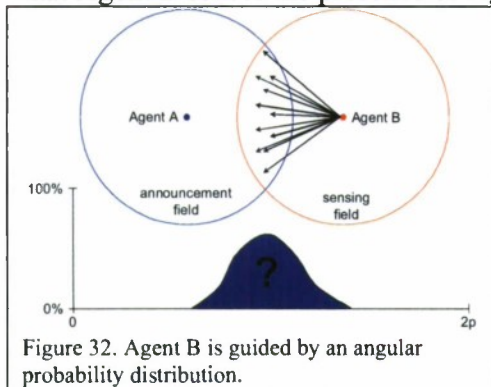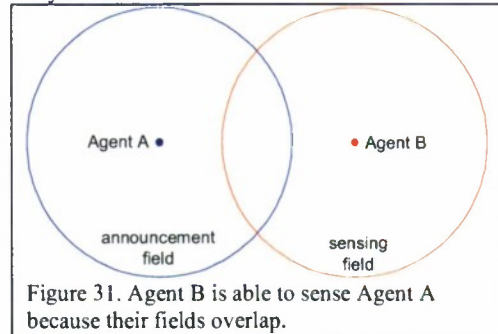#### 9.3.5.2.2 Abstraction 2: Circular Sensing Fields

As in abstraction one, we can assume that an infinite number of ghosts will sample the presence of pheromone concentrations at all locations within the fixed radius of N*stepLength around their avatar. Therefore, the avatar will have complete knowledge of the distribution of pheromone concentrations from other avatars within this "sensing radius" without actually deploying ghosts.

Conclusion: We assume the ability of an avatar to sense other avatars' fields if they are within the fixed radius of the "sensing" field.

9.3.5.2.3 Abstraction 3: Avatar Move by Angular Probability Distribution

After abstraction one and two, we no longer have ghosts in our model. Rather, there are only agents (former avatars) that are able to sense the presence of other agents if the "announcement" field of one agent overlaps with the "sensing" field of the other. So, in Figure 31, Agent B's sensing field overlaps with Agent A's announcement field and therefore, we assume that Agent B senses the presence of Agent A.

Figure 31. Agent B is able to sense Agent A because their fields overlap.

Figure 32. Agent B is guided by an angular probability distribution.

In our Probabilistic Walker Model, we used the sampling process of a finite number of ghosts performing random walks around their avatar to determine the direction for the next step of the avatar. This process has a strong noise component as ghosts from avatar A would deposit pheromones not only along the direct path to avatar B, but anywhere in the circular area of the announcement field and, within the overlap of the sensing field, ghosts from avatar B would encounter these deposits. To model this noise component, we need to specify an angular movement probability distribution as a function of the overlap of the announcement field with the sensing field.

Conclusion: In each step, an agent calculates a probability distribution over all possible angles (0-2Pi) that is dependent on the overlap of its sensing field with the announcement fields of other agents. The agent then samples this probability distribution to determine the direction of its next fixed-length step.

9.3.5.2.4 Field-Overlap Probabilities

For our "Mean-Field" approximation of the Probabilistic Walker Model, we need to describe the angular probability distribution that derives from the overlap of an agent's sensing field with another agent's announcement field. For now we assume that the radius of the sensing field is equal to the radius of the announcement field. The "shape" of this overlap is a function of the distance of the agents:

1) If the agents are too far away, then the fields don't overlap and the angular probability distribution is uniform (agent performs random walk). Otherwise, the probabilities are non-uniform and, in particular, the heading angle pointing directly towards the other agent will always have the highest probability (except in the case of zero distance).

2) If the agents are more than one field radius away from each other (as in Figure 31), then all angles pointing away from the other agent have zero probabilities. Furthermore, depending on the actual distance of the agents, some of the angles that would still move the agents closer to each other (but not along the direct path) also have zero probabilities.

3) If the agents are less than one field radius away from each other, then the announcement field of agent A actually covers the location of agent B and even reaches behind it. In this case, the angular probabilities are all non-zero, which means that the agent has a non-zero probability to actually move away from the other agent.
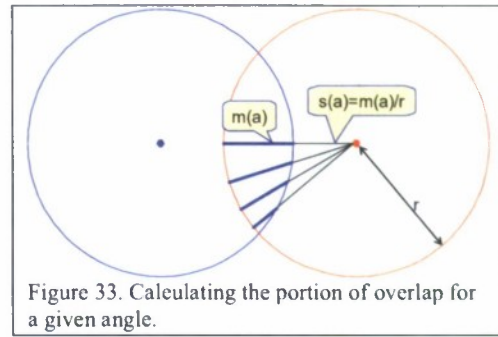

Figure 33. Calculating the portion of overlap for a given angle.

4) If the agents share the exact same location, then all angles are of equal selection probability.

Based on these observations, we defined the following geometric method to calculate the probability distribution. For any angle *(a)* around agent B, consider the line that connects the agent's location with the edge of its sensing radius in the direction of the angle. Then determine the length *m(a)* of the portion of the line that is overlapped by the announcement field of agent A – this portion may be zero. Let *s(a)* denote the relative length of *m(a)* compared to the radius of the field (*s(a)=m(a)/r*). Finally, the probability *p(a)* for agent B to move in direction *a* is *s(a)* normalized for all possible angles.

Figure 34 shows the relative degree of overlap (*s(a)*) for all angles as a function of the distance of two agents with a field radius of 10. When the distance is exactly twice the field radius (20), there is only one direction that has non-zero overlap. As the distance shrinks, the range of angles with non-zero overlap widens until, at a distance equal to the field radius, suddenly all angles report non-zero announcement field presence.

As we normalize these *s(a)*-values for all possible angles (Figure 35), we find that the widening of the range of available angles quickly leads to a "dilution" of the guidance provided by the field. In other words, the likelihood for agent B to move directly towards agent A is diminishing rapidly.
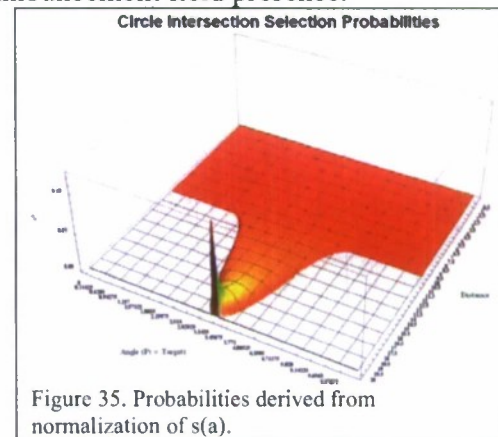

Figure 35. Probabilities derived from normalization of s(a).
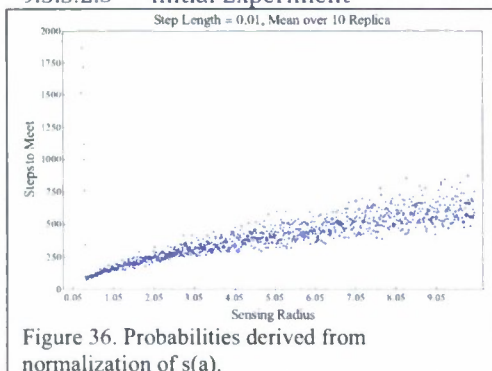
9.3.5.2.5   Initial Experiment

Figure 36. Probabilities derived from normalization of s(a).

We executed an initial experiment with the Mean-Field Walker Model, where we systematically varied the field radius of the agents. Figure 36 shows the time it takes two
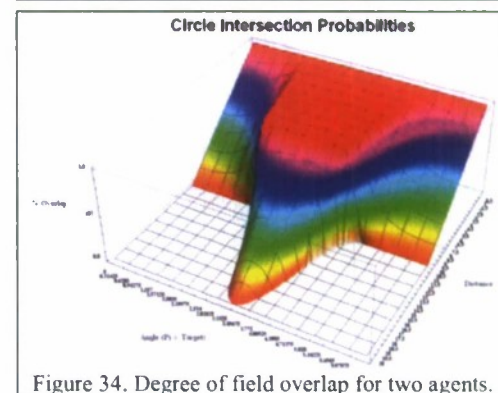

Figure 34. Degree of field overlap for two agents.

agents to meet (mean over ten replications with different random seeds) for a fixed agent step length (0.01) and varying sensing radius. As in the Probabilistic Walker Model (Figure 29), the performance peaks (time to meet dips) at an optimal sensing radius. For smaller radii, the agents take too long to get "in range" (detect each other's field), while for larger radii, the agents are confused by the significant overlap of their fields.

10. **Deliverables, Personnel, Awards, Publications, Presentations, and Patents**

Deliverable	Date
Final Technical Report (this document) with SF298	June 19, 2009

Table 1. Deliverables

ONR C-IED STIFLE Final Report

	Total #	Name	Organization
PIs	1	Dr. Sven Brueckner	TechTeam Government Solutions
Project Staff	6	Ted Belding	TechTeam Government Solutions
		Dr. Rainer Hilscher	TechTeam Government Solutions
		Dr. Van Parunak	TechTeam Government Solutions
		Liz Downs	TechTeam Government Solutions
		Steve Brophy	TechTeam Government Solutions
		Rob Bisson	TechTeam Government Solutions
Subcontractors	2	Dr. Keith Decker	University of Delaware
		Dr. Robert Savit	Savit Research Associates and University of Michigan

Table 2. Personnel Information

Name	Award (Year Received)	Prize (Year Received)	Recognition (Year Received)
N/A			

Table 3. Awards/Prizes/Recognitions

Name	Publication (Date)	Conference Presentation (Date)	Patent (Date)
H. Van Dyke Parunak, Theodore C. Belding, and Sven Brueckner.	Prediction Horizons in Polyagent Models. In *Proceedings of Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS07)*, Honolulu, HI, 2007.		
Sven Brueckner, Theodore C. Belding, Robert Bisson, Elizabeth Downs, and H. Van Dyke Parunak	Swarming Polyagents Executing Hierarchical Task Networks. In *Proceedings of Third IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO2009)*, San Francisco, CA, 2009 (forthcoming).		
H. Van Dyke Parunak, Theodore Belding, Robert Bisson, Sven Brueckner, Elizabeth Downs, Rainer Hilscher, and Keith Decker.	Stigmergic Modeling of Hierarchical Task Networks. In *Proceedings of the Tenth International Workshop on Multi-Agent-Based Simulation (MABS 2009, at AAMAS 2009)*, Budapest, Hungary, (forthcoming), 2009.		
H. Van Dyke Parunak, Theodore Belding, Rainer Bisson, Sven Brueckner, Elizabeth Downs, and Rainer Hilscher.	Stigmergic Reasoning over Hierarchical Task Networks. In *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 1195–1196.		

Table 4. Publications, Conference Presentations and Patents

11. References:

[1] M. Boddy, B. Horling, J. Phelps, R. P. Goldman, R. Vincent, A. C. Long, B. Kohout, and R. Maheswaran. C_TAEMS Language Specification, Version 2.02. DARPA, Arlington, VA, 2006.

[2] S. Brueckner. Return from the Ant: Synthetic Ecosystems for Manufacturing Control. Dr.rer.nat. Thesis at Humboldt University Berlin, Department of Computer Science, 2000.

[3] W. Chen and K. Decker. The Analysis of Coordination in an Information System Application – Emergency Medical Services. In P. Bresciani, P. Giorgini, B. Henderson-Sellers, and M. Winiko, Editors, Agent-Oriented Information Systems, vol. 3508, LNAI, pages 36-51. Springer, New York, NY, 2005.

[4] K. Decker and J. Li. Coordinating mutually exclusive resources using GPGP. Journal of Autonomous Agents and Multi-Agent Systems, 3(2):133-158, 2000.

[5] B. Horling, V. Lesser, R. Vincent, T. Wagner, A. Raja, S. Zhang, K. Decker, and A. Garvey. The TAEMS White Paper. 2004.

[6] R. T. Maheswaran, P. Szekely, M. Becker, S. Fitzpatrick, G. Gati, J. Jin, R. Neches, N. Noori, C. Rogers, R. Sanchez, K. Smyth, and C. Vanbuskirk. Predictability & Criticality Metrics for Coordination in Complex Environments. In Proceedings of Seventh International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2008), Estoril, PT, 2008.

[7] D. J. Musliner, E. H. Durfee, JianhuiWu, D. A. Dolgov, R. P. Goldman, and M. S. Boddy. Coordinated Plan Management Using Multi-agent MDPs. In Proceedings of AAAI Spring Symposium on Distributed Plan and Schedule Management, Palo Alto, CA, 2006.

[8] H. V. D. Parunak, T. Belding, R. Bisson, S. Brueckner, E. Downs, R. Hilscher, and K. Decker. Stigmergic Modeling of Hierarchical Task Networks. In *Proceedings of the Tenth International Workshop on Multi-Agent-Based Simulation (MABS 2009, at AAMAS 2009)*, Budapest, Hungary, (forthcoming), 2009.

[9] H. V. D. Parunak and S. Brueckner. Concurrent Modeling of Alternative Worlds with Polyagents. In Proceedings of the Seventh International Workshop on Multi-Agent-Based Simulation (MABS06, at AAMAS06), Hakodate, Japan, Springer, 2006.

[10] S. F. Smith, A. Gallagher, T. Zimmerman, L. Barbulescu, and Z. Rubinstein. Distributed Management of Flexible Times Schedules. In Proceedings of Sixth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2007), Honolulu, Hawaii, pages 472-479, 2007.